Trzynaście Wykładów z Matematyki Obliczeniowej

Leszek Plaskota
Instytut Matematyki Stosowanej i Mechaniki
Uniwersytet Warszawski

Spis treści

Rozdział 1

Specyfika obliczeń numerycznych

Matematyka obliczeniowa jest dziedziną wiedzy zajmującą się problemami obliczeniowymi i konstrukcją algorytmów rozwiązywania zadań matematycznych.

Aby w ogóle mówić w problemie obliczeniowym, musimy najpierw

- określić dane początkowe i cel obliczeń, czyli dokładnie sformułować zadanie,
- określić środki obliczeniowe dzięki którym chcemy osiągnąć cel, czyli zdefiniować *model obliczeniowy.*

1.1 Zadania numeryczne, przykłady

Sformułowanie zadania polega na sprecyzowaniu tego, co mamy i co chcemy uzyskać. Formalnie polega to na zdefiniowaniu przestrzeni danych F, przestrzeni wyników G i wskazaniu zależności $f \mapsto g$, gdzie $f \in F$ i $g \in G$, między danymi a wynikami. Zależność tę repezentuje *operator rozwiązania*

$$S : F \longrightarrow G.$$

Będziemy zainteresowani przede wszystkim zadaniami *numerycznymi.* Są to zadania, dla których wynikiem g jest zawsze skończony

ciąg liczb rzeczywistych, czyli

$$G \subset \cup_{n=0}^{\infty} \mathbf{R}^n$$

(przyjmujemy przy tym, że $\mathbf{R}^0 = \mathit{pusty}$). Zbiór danych może być w zasadzie dowolny, ale my będziemy dla uproszczenia rozpatrywać tylko zadania dla których $F \subset \mathbf{R}^m$, albo ogólniej, dla których F jest pewną klasą funkcji zdefiniowanych na ustalonym zbiorze $D \subset \mathbf{R}^d$.

Przykład 1.1 *(Równanie kwadratowe.)* Załóżmy, że chcemy obliczyć wszystkie rzeczywiste pierwiastki równania

$$x^2 - 2p + q = 0,$$

dla danych liczb rzeczywistych p i q. Wtedy $F = \mathbf{R}^2$, $G = \mathbf{R}^0 \cup \mathbf{R}^1 \cup \mathbf{R}^2$, oraz

$$S(p,q) = \begin{cases} \mathit{pusty} & \Delta < 0, \\ p & \Delta = 0, \\ (p+\sqrt{\Delta}, p-\sqrt{\Delta}) & \Delta > 0, \end{cases} \tag{1.1}$$

gdzie $\Delta = p^2 - q$.

Przykład 1.2 *(Układ równań liniowych.)* Rozpatrzmy zadanie rozwiązywania układu równań liniowych

$$A\vec{x} = \vec{b},$$

dla nieosobliwej macierzy $A = (a_{i,j})_{i,j=1}^n$ i wektora $\vec{b} = (b_j)_{j=1}^n$. Wtedy

$$F = \{ (A, \vec{b}) \in \mathbf{R}^{n^2+n} : \ \det A \neq 0 \},$$

$G = \mathbf{R}^n$, oraz

$$S(A, \vec{b}) = A^{-1}\vec{b}.$$

Przykład 1.3 *(Całkowanie.)* Dla danej funkcji ciągłej $f : [a,b] \to \mathbf{R}$, chcemy obliczyć całkę oznaczoną

$$I(f) = \int_a^b f(x)\,dx.$$

W tym przypadku $G = \mathbf{R}$, ale dane stanowi pewna klasa funkcji ciągłych określonych na odcinku $[a,b]$, tzn. $F \subset \mathbf{C}([a,b])$. Oczywiście S jest operatorem całkowania, $S = I$.

1.2 Model obliczeniowy

Aby zdefiniować nasz model obliczeniowy, posłużymy się pojęciem *programu*. Zastosujemy przy tym notację podobną do tej z języka programowania *Pascal*.

Program składa się z *deklaracji*, czyli opisu obiektów, których będziemy używać, oraz z *poleceń* (*instrukcji*), czyli opisu akcji, które będziemy wykonywać.

Dostępnymi obiektami są *stałe* i *zmienne* typu całkowitego (*integer*), rzeczywistego (*real*), albo logicznego (*Boolean*). Zmienne mogą być grupowane w wektory albo tablice.

Polecenia dzielimy na proste i złożone. Poleceniem prostym jest

- PODSTAWIENIE

$$z := \mathcal{W};$$

gdzie z jest zmienną, a $\mathcal{W}$ jest *wyrażeniem* o wartościach tego samego typu co z.

Wyrażeniem jest pojedyncza stała lub zmienna, albo złożenie skończonej liczby *operacji elementarnych* na wyrażeniach. Operacje elementarne to:

arytmetyczno–arytmetyczne: $x \mapsto -x$, $(x,y) \mapsto x+y$, $(x,y) \mapsto x-y$, $(x,y) \mapsto x*y$, $(x,y) \mapsto x/y, y \neq 0$, gdzie x,y są stałymi lub zmiennymi liczbowymi,

arytmetyczno–logiczne: $(x,y) \mapsto x<y$, $(x,y) \mapsto x \leq y$, $(x,y) \mapsto x=y$, $(x,y) \mapsto x \neq y$, gdzie x,y są stałymi lub zmiennymi liczbowymi,

logiczno–logiczne: $p \mapsto not\, p$, $(p,q) \mapsto p\, and\, q$, $(p,q) \mapsto p\, or\, q$, gdzie p,q są stałymi lub zmiennymi logicznymi.

Dla niektórych zadań wygodnie jest (a czasem koniecznie) uzupełnić ten zbiór o dodatkowe operacje, takie jak obliczanie wartości niektórych funkcji standardowych ($\sqrt{\;}, \cos(), \sin(), \exp(), \log()$, itp.),

czy nawet funkcji bardziej skomplikowanych. Na przykład, zastosowanie “szkolnych” wzorów na obliczanie pierwiatków równania kwadratowego z Przykładu 1.1 byłoby niemożliwe, gdyby pierwiastkowanie było niemożliwe. Należy jednak przy tym pamiętać, że w praktyce funkcje standardowe (o ile są dopuszczalne) są obliczane używając czterech podstawowych operacji arytmetycznych.

Mamy trzy podstawowe polecenia złożone.

- Warunkowe

$$\textbf{if} \quad \mathcal{W} \quad \textbf{then} \quad \mathcal{A}_1 \quad \textbf{else} \quad \mathcal{A}_2;$$

gdzie W jest wyrażeniem o wartościach logicznych, a $\mathcal{A}_1$ i $\mathcal{A}_2$ są poleceniami, przy czym dopuszczamy polecenia puste.

- Powtarzane

$$\textbf{while} \quad \mathcal{W} \quad \textbf{do} \quad \mathcal{A};$$

gdzie W jest wyrażeniem o wartościach logicznych, a $\mathcal{A}$ jest poleceniem.

- Kombinowane

$$\textbf{begin} \quad \mathcal{A}_1;\ \mathcal{A}_2;\ \ldots\ ;\ \mathcal{A}_n \quad \textbf{end};$$

gdzie $\mathcal{A}_j$ są poleceniami.

Na podstawie tych trzech poleceń można tworzyć inne, takie jak pętle **for** i **repeat**, czy **case**, itd.

Mamy też dwa szczególne polecenia, które odpowiadają za ”wejście” i ”wyjście”.

- Wprowadzanie danych

$$\mathcal{IN}(x,t);$$

gdzie x jest zmienną rzeczywistą, a t “adresem” pewnego funkcjonału $L : F \to \mathbf{R}$ należącym to pewnego zbioru T. W wyniku wykonania tego polecenia w zmiennej x zostaje umieszczona wartość $L_t(f)$.

Polecenie to pozwala zdobyć *informację* o danej f. Jeśli $F = \mathbf{R}^n$ to zwykle mamy $T = \{1, 2, \ldots, n\}$ i $L_i(f) = f_i$, co w praktyce odpowiada

wczytaniu i-tej współrzędnej wektora danych. W szczególności, ciąg poleceń $\mathcal{IN}(x[i], i)$, $i = 1, 2, \ldots, n$, pozwala uzyskać pełną informację o f. Jeśli zaś F jest pewną klasą funkcji $f : [a, b] \to \mathbf{R}$, to możemy mieć np. $T = [a, b]$ i $L_t(f) = f(t)$. W tym przypadku, wykonanie polecenia $\mathcal{IN}(x, t)$ odpowiada w praktyce skorzystaniu ze specjalnej procedury (albo urządzenia zewnętrznego) obliczającej (mierzącego) wartość funkcji f w punkcie t.

- Wyprowadzanie wyników

$$\mathcal{OUT}(\mathcal{W});$$

gdzie $\mathcal{W}$ jest wyrażeniem o wartościach rzeczywistych. Polecenie to pozwala "wskazać" kolejną współrzędną wyniku.

Zakładamy, że na początku procesu obliczeniowego wartości wszystkich zmiennych są nieokreślone, oraz że dla dowolnych danych wykonanie programu wymaga wykonania skończonej liczby operacji elementarnych. Wynikiem obliczeń jest skończony ciąg liczb rzeczywistych (albo *puste*), którego kolejne współrzędne pokazywane są poleceniem $\mathcal{OUT}$.

1.3 Arytmetyka fl_ν

Przedstawiony model obliczeniowy jest modelem idealistycznym, tzn. zakłada on, że wszystkie operacje są wykonywane bezbłędnie. Dlatego w tym przypadku będziemy mówić o *arytmetyce idealnej*. W praktyce jednak, np. wykonując obliczenia na maszynie cyfrowej, operacje arytmetyczne na liczbach rzeczywistych wykonywane są z pewnym błędem. Matematycznym modelem arytmetyki maszyny cyfrowej jest *arytmetyka fl_ν* (albo arytmetyka *zmiennoprzecinkowa*), którą teraz przedstawimy.

Dowolną liczbę rzeczywistą $x \neq 0$ można przedstawić w postaci

$$x = s \cdot 10^c \cdot m,$$

gdzie $s \in \{-1, 1\}$ jest znakiem, $c \in \mathbf{Z}$ *cechą*, a $m \in [0.1, 1.0)$ *mantysą* liczby x. Zauważmy, że taki rozkład jest jednoznaczny i odpowiada przesuwaniu przecinka w rozwinięciu dziesiętnym liczby do pierwszej cyfry

znaczącej, tj. różnej od zera. Mantysa ma w ogólności nieskończenie wiele cyfr c_j w swoim rozwinięciu dziesiętnym,

$$m = \sum_{j=1}^{\infty} c_j 10^{-j},$$

gdzie $c_j \in \{0, 1, 2, \ldots, 9\}$ i $c_1 \neq 0$. Jako taka nie może więc być zapamiętana dokładnie w pamięci maszyny cyfrowej. Zakładając, że pamiętamy tylko t cyfr znaczących i ostatnią cyfrę zaokrąglamy, zamiast m pamiętamy

$$m_t = \sum_{j=1}^{t} c_j 10^{-j} + d_t 10^{-j},$$

gdzie

$$d_t = \begin{cases} 0 & \text{gdy } 0 \leq c_{t+1} \leq 4\}, \\ 1 & \text{gdy } 5 \leq c_{t+1} \leq 9\}. \end{cases}$$

Stąd x będzie reprezentowana przez liczbę

$$rd(x) = s \cdot 10^c \cdot m_t.$$

Między liczbą x ($x \neq 0$) a jej reprezentacją $rd(x)$ zachodzi więc nierówność

$$\frac{|x - rd(x)|}{|x|} = \frac{|m - m_t|}{|m|} \leq \frac{5 \cdot 10^{-(t+1)}}{10^{-1}} = 5 \cdot 10^{-t},$$

charakteryzująca dokładność arytmetyki. Liczby rzeczywiste są więc w arytmetyce fl_ν reprezentowane z dokładnością *względną* $\nu = 5 \cdot 10^{-t}$. (We współczesnych maszynach cyfrowych w pojedynczej precyzji mamy co najmniej $t = 8$.) Zauważmy, że t jest największą spośród liczb naturalnych j spełniających bierówność $rd(1 + 10^{-j}) > 1$.

Ostatnią nierówność wygodnie jest zapisać w równoważny sposób jako

$$rd(x) = x(1 + \varepsilon), \qquad \text{gdzie} \quad |\varepsilon| \leq \nu.$$

W arytmetyce fl_ν zakładamy ponadto, że działania arytmetyczne na liczbach rzeczywistych (a raczej na ich reprezentacjach) są wykonywane dokładnie i tylko wynik jest zaokrąglany. Mamy więc

$$fl_\nu(x \,\square\, y) = rd\,(\,rd(x) \,\square\, rd(y)\,)\,,$$

gdzie $\Box \in \{+,-,*,/\}$, Ogólniej, jeśli $\mathcal{W}_1$ i $\mathcal{W}_2$ są wyrażeniami o wartościach rzeczywistych, to dla dowolnych wartości zmiennych

$$fl_\nu(\mathcal{W}_1 \Box \mathcal{W}_2) \;=\; rd\,(fl_\nu(\mathcal{W}_1) \Box fl_\nu(\mathcal{W}_2))\,.$$

Zwykle dla prostoty będziemy również zakładać podobną zależność dla niektórych funkcji standardowych, o ile należą one do zbioru operacji elementarnych (chociaż w rzeczywistości są one obliczane przez procedury używające czterech podstawowych operacji arytmetycznych). I tak będziemy mieć np.

$$\begin{aligned} fl_\nu\Big(\sqrt{\mathcal{W}}\Big) &= \Big(\sqrt{fl_\nu(\mathcal{W})}\Big)\,(1+\beta_1), \\ fl_\nu(\mathcal{W}) &= (\cos(fl_\nu(\mathcal{W})))\,(1+\beta_2), \end{aligned}$$

gdzie $|\varepsilon_j| \le \nu$, oraz $\beta_j \le K_j\nu$ i K_j są "niewielkimi" stąłymi.

Podobnie, jeśli $\triangle$ jest operatorem porównania, $\triangle \in \{<,\le,=,\ne\}$, to wartością wyrażenia logicznego $\mathcal{W}_1 \triangle \mathcal{W}_2$ w fl_ν jest dokładna wartość wyrażenia $fl_\nu(\mathcal{W}_1)\triangle fl_\nu(\mathcal{W}_2)$.

Uwagi i uzupełnienia

U. 1.1 W maszynie cyfrowej cecha c liczby rzeczywistej nie może oczywiście mieć dowolnie dużej wartości bezwzględnej, $|c| \le c_{\max}$. Powoduje to powstanie zjawiska *nadmiaru* gdy $c > c_{\max}$, oraz zjawiska *niedomiaru* gdy $c < -c_{\max}$. W pierwszym przypadku liczba jest tak duża (co do modułu), że nie zawiera się w przedziale liczb reprezentowalnych, a w drugim jest tak mała, że musi być reprentowana przez zero, przy czym błąd względny reprezentacji wynosi wtedy 1 a nie ν. W dalszych rozważaniach zjawiska nadmiaru i niedomiaru będziemy dla uproszczenia zaniedbywać.

U. 1.2 W następnych rozdziałach często będziemy się posługiwać normami wektorów $\vec{x} = (x_j)_{j=1}^n \in \mathbf{R}^n$ i macierzy $A = (a_{i,j})_{i,j=1}^n \in \mathbf{R}^{n\times n}$, a w szczególności błędami reprezentacji wektorów i macierzy w normie. Najczęściej używanymi normami wektorowymi będą normy p-te,

$$\|\vec{x}\| \;=\; \|\vec{x}\|_p \;=\; \left(\sum_{j=1}^n |x_j|^p\right)^{1/p}, \qquad 1 \le p < +\infty,$$

oraz

$$\|\vec{x}\|_\infty \;=\; \lim_{p\to+\infty} \|\vec{x}\|_p \;=\; \max_{1\le j\le n} |x_j|.$$

Normą macierzową jest norma euklidesowa (albo Frobeniusa)

$$\|A\|_E \;=\; \sqrt{\sum_{i,j=1}^{n} |a_{i,j}|^2},$$

a także normy *indukowane* przez normy wektorowe (np. przez normy p-te)

$$\|A\| \;=\; \sup_{x\neq 0} \frac{\|A\vec{x}\|}{\|\vec{x}\|} \;=\; \sup_{\|x\|=1} \|A\vec{x}\|.$$

Jeśli norma macierzowa jest indukowana przez normę wektorową, to dla dowolnego wektora mamy

$$\|A\vec{x}\| \;\le\; \|A\|\|\vec{x}\|.$$

U. 1.3 Przypomnijmy, że w przestrzeniach liniowych skończenie wymiarowych (a więc także w $\mathbf{R}^n$ i w przestrzeni macierzy wymiaru $n\times n$) każde dwie normy są równoważne. To znaczy, że jeśli mamy dwie normy $\|\cdot\|$ i $\|\cdot\|'$ w przestrzeni skończenie wymiarowej X, to istnieją stałe $0 < K_1 \le K_2 < \infty$ takie, że

$$K_1\,\|x\| \;\le\; \|x\|' \;\le\; K_2\,\|x\|, \qquad \forall x\in X.$$

W szczególności dla $\vec{x}\in\mathbf{R}^n$ mamy

$$\|\vec{x}\|_\infty \;\le\; \|\vec{x}\|_1 \;\le\; n\,\|\vec{x}\|_\infty, \tag{1.2}$$

$$\|\vec{x}\|_\infty \;\le\; \|\vec{x}\|_2 \;\le\; \sqrt{n}\,\|\vec{x}\|_\infty, \tag{1.3}$$

$$\frac{1}{\sqrt{n}}\,\|\vec{x}\|_1 \;\le\; \|\vec{x}\|_2 \;\le\; \|\vec{x}\|_1, \tag{1.4}$$

a dla $A = (a_{i,j})_{i,j=1}^n$ mamy

$$\|A\|_2 \;\le\; \|\,|A|\,\|_2 \;\le\; \|A\|_E \;\le\; \sqrt{n}\,\|A\|_2, \tag{1.5}$$

gdzie $|A| = (|a_{i,j}|)_{i,j=1}^n$.

Ćwiczenia

Ćw. 1.1 Podać przykłady liczb x i y, które są dokładnie reprezentowane w fl_ν, ale $fl_\nu(x\Box y) \neq x\Box y$, gdzie $\Box \in \{+,-,*,/\}$.

Ćw. 1.2 Uzasadnić, że jeśli $x \leq y$ to $rd(x) \leq rd(y)$ oraz $fl_\nu(x/y) \leq 1$. Podać przykład wyrażenia $\mathcal{W}$ takiego, że dla pewnych wartości zmiennych $\mathcal{W} < 0$, ale $fl_\nu(\mathcal{W}) > 0$.

Ćw. 1.3 Jak zabezpieczyć się przed powstaniem nadmiaru albo niedomiaru przy obliczaniu wyrażenia $\sqrt{x^2+y^2}$, gdy x i y leżą w przedziale liczb reprezentowalnych w fl_ν, ale x^2 lub y^2 nie?

Ćw. 1.4 Uzasadnić nierówności (1.2)–(1.5).

Ćw. 1.5 Pokazać, że dla macierzy $A = (a_{i,j})_{i.j=1}^n \in \mathbf{R}^{n\times n}$ mamy

$$\|A\|_\infty \;=\; \max_{1\leq i\leq n} \sum_{j=1}^n |a_{i,j}|$$

oraz

$$\|A\|_1 \;=\; \|A^T\|_\infty \;=\; \max_{1\leq j\leq n} \sum_{i=1}^n |a_{i,j}|.$$

Ćw. 1.6 Dla wektora $\vec{x} = (x_j)_{j=1}^n \in \mathbf{R}^n$, niech $rd(\vec{x}) = (rd(x_j))_{j=1}^n$. Pokazać, że

$$\|\vec{x} - rd(\vec{x})\|_p \;\leq\; \nu\,\|\vec{x}\|_p$$

dla $1 \leq p \leq \infty$.

Ćw. 1.7 Dla macierzy $A = (a_{i,j})_{i,j=1}^n$, niech $rd(A) = (rd(a_{i,j}))_{i,j=1}^n$. Pokazać, że

$$\|A - rd(A)\|_p \;\leq\; \nu\,\|A\|_p,$$

dla $p = 1, \infty$, oraz

$$\|A - rd(A)\|_2 \;\leq\; \|A - rd(A)\|_E \;\leq\; \nu\,\|A\|_E \;\leq\; \sqrt{n}\,\nu\,\|A\|_2.$$

æ

Rozdział 2

Algorytmy i ich własności

Algorytm to dokładnie określona i dozwolona w danym modelu obliczeniowym sekwencja akcji, pozwalająca na rozwiązanie danego zadania (w sposób dokładny lub przybliżony).

2.1 Rozkład algorytmu względem informacji

Z każdym algorytmem związany jest operator

$$\Phi : F \longrightarrow G,$$

taki że $\Phi(f)$ jest wynikiem działania algorytmu w arytmetyce idealnej dla danej f.

Zauważmy, że wynik $\Phi(f)$ działania algorytmu nie zależy bezpośrednio od f, ale raczej od *informacji* o f (uzyskanej dzięki poleceniu $\mathcal{IN}$). Informacja ta może być *pełna* albo tylko *częściowa*. Informacja jest pełna gdy, np. $f = (f_1, \ldots, f_n) \in \mathbf{R}^n$ i wczytamy wszystkie współrzędne f_i. Informacja może być częściowa, gdy f jest funkcją. Wtedy wiele danych może posiadać tę samą informację, co łatwo zaobserwować na przykładzie zadania całkowania.

Niech $N : F \to \cup_{n=0}^{\infty} \mathbf{R}^n$ będzie *operatorem informacji*, tzn.

$$N(f) \;=\; (y_1, y_2, \ldots, y_n)$$

jest informacją o f zebraną przy idealnej realizacji algorytmu. Zauważmy, że nformacja jest pełna gdy N jest przekształceniem różnowartościowym, tzn. jeśli $f_1 \neq f_2$ implikuje $N(f_1) \neq N(f_2)$. W przeciwnym przypadku mamy do czynienia z informacją częściową.

Każdy algorytm Φ może być przedstawiony jako złożenie operatora informacji i pewnego operatora $\varphi : N(F) \to G$ zdefiniowanego równością

$$\varphi\,(N(f)) \;=\; \Phi(f).$$

Zauważmy, że w przypadku informacji częściowej zwykle nie istnieje algorytm dający dokładne rozwiązanie zadania dla każdej danej $f \in F$, ponieważ dla danych o tej samej informacji mogą istnieć różne rozwiązania.

2.2 Problem wyboru algorytmu

Wybór algorytmu jest najistotniejszą częścią całego procesu numerycznego rozwiązywania zadania. Kierujemy się przy tym przede wszystkim następującymi kryteriami:

- dokładnością algorytmu,
- złożonością algorytmu,
- własnościami numerycznymi algorytmu.

Przez dokładność algorytmu rozumiemy różnicę między rozwiązaniem dokładnym $S(f)$, a rozwiązaniem $\Phi(f)$ dawanym przez algorytm w arytmetyce idealnej. Jeśli $\Phi(f) = S(f)$, $\forall f \in F$, to algorytm nazywamy *dokładnym*.

Mówiąc o złożoności, mamy na myśli złożoność pamięciową (zwykle jest to liczba stałych i zmiennych używanych przez algorytm), jak również złożoność obliczeniową. Na złożoność obliczeniową algorytmu dla danej f składa się koszt uzyskania infomacji $y = N(f)$ (zwykle jest on proporcjonalny do liczby wywołań polecenia $\mathcal{IN}$), oraz koszt *kombinatoryczny* przetworzenia tej informacji, aż do uzyskania wyniku $\varphi(y)$. Koszt kombinatoryczny zwykle mierzymy liczbą operacji arytmetycznych wykonywanych przez algorytm.

Przez własności numeryczne algorytmu rozumiemy jego własności przy realizacji w arytmetyce fl_ν. Temu ważnemu tematowi poświęcimy teraz osobny paragraf.

2.3 Numeryczna poprawność algorytmu

Pożądane jest, aby algorytm dawał "dobry" wynik zarówno w arytmetyce idealnej, jak i w arytmetyce fl_ν. Niestety, jak zobaczymy, nie zawsze jest to możliwe. Nawet jeśli algorytm jest dokładny to w wyniku jego realizacji w fl_ν możemy otrzymać wynik $fl_\nu(\Phi(f))$ daleko odbiegający od $S(f)$. W szczególności, prawie zawsze mamy

$$S(f) \neq fl_\nu(\Phi(f)).$$

Zauważmy również, że o ile z reguły znamy dokładne zachowanie się algorytmu w arytmetyce idealnej dla danej informacji, to nie można tego samego powiedzieć o jego zachowaniu się w arytmetyce fl_ν. W związku z tym powstaje pytanie, jak kontrolować błąd algorytmów wynikający z błędów zaokrągleń i jakie algorytmy uznamy za te o najwyższej jakości numerycznej.

Istnienie błędów reprezentacji liczb rzeczywistych powoduje, że informacja $y = N(f)$ o danej f nie jest w ogólności reprezentowana dokładnie. Znaczy to, że zamiast na informacji dokładnej, dowolny algorytm będzie operować na informacji *nieco zaburzonej* y_ν, tzn. zaburzonej na poziomie błędu reprezentacji. Tak samo wynik dawany przez algorytm będzie w ogólności zaburzony na poziomie błędu reprezentacji. W najlepszym więc wypadku wynikiem działania algorytmu w fl_ν będzie $(\varphi(y_\nu))_\nu$ zamiast $\varphi(y)$. Algorytmy dające tego rodzaju wyniki uznamy za posiadające najlepsze własności numeryczne w arytmetyce fl_ν i nazwiemy numerycznie poprawnymi.

Dokładniej, powiemy, że ciąg rzeczywisty $a_\nu = (a_{\nu,1}, \ldots, a_{\nu,n})$ (a właściwie rodzina ciągów $\{a_\nu\}_\nu$) jest *nieco zaburzonym* ciągiem $a = (a_1, \ldots, a_n)$, jeśli istnieje stała K taka, że dla wszystkich dostatecznie małych ν zachodzi

$$|a_{\nu,j} - a_j| \leq K\,\nu\,|a_j|, \qquad 1 \leq j \leq n, \tag{2.1}$$

albo ogólniej

$$\|a_\nu - a\| \;\le\; K\,\nu\,\|a\|, \tag{2.2}$$

gdzie $\|\cdot\|$ jest pewną normą w $\mathbf{R}^n$. W pierwszym przypadku mówimy o zaburzeniu "po współrzędnych", a w drugim o zaburzeniu w normie $\|\cdot\|$.

Zauważmy, że niewielkie zaburzenia po współrzędnych pociągają za sobą niewielkie zaburzenia w normie. Rzeczywiście, jeśli (2.1) to również

$$\|a_\nu - a\|_\infty \;=\; \max_{1\le j\le n} |a_{\nu,j} - a_j| \;\le\; K\,\nu \max_{1\le j\le n} |a_j| \;=\; K\,\nu\,\|a\|_\infty,$$

i korzystając z faktu, że w przestrzeni skończenie wymiarowej wszystkie normy są równoważne otrzymujemy dla pewnych stałych K_1 i K_2

$$\|a_\nu - a\| \;\le\; K_1\|a_\nu - a\|_\infty \;\le\; K_1 K\,\nu\,\|a\|_\infty \;\le\; K_2 K_1 K\,\nu\,\|a\|,$$

czyli nierówność (2.2) ze stałą K_2K_1K zamiast K.

Definicja 2.1 Algorytm Φ rozwiązywania zadania nazywamy *numerycznie poprawnym* w zbiorze danych $F_0 \subset F$, jeśli dla każdej danej $f \in F_0$ wynik $fl_\nu(\Phi(f))$ działania algorytmu w arytmetyce fl_ν można zinterpretować jako nieco zaburzony wynik algorytmu w arytmetyce idealnej dla nieco zaburzonej informacji $y_\nu = (N(f))_\nu \in N(F)$ o f, przy czym poziom zaburzeń nie zależy od f.

Formalnie znaczy to, że istnieją stałe K_1, K_2, oraz $\nu_0 > 0$ takie, że spełniony jest następujący warunek. Dla dowolnej $\nu \le \nu_0$ oraz informacji $y \in N(F_0)$ można dobrać $y_\nu \in N(F)$ oraz $(\varphi(y_\nu))_\nu$ takie, że

$$\|y_\nu - y\| \;\le\; K_1\,\nu\,\|y\|,$$

$$\|\,(\varphi(y_\nu))_\nu - \varphi(y_\nu)\| \;\le\; K_2\,\nu\,\|\varphi(y_\nu)\|,$$

oraz

$$fl_\nu\,(\Phi(f)) \;=\; fl_\nu\,(\varphi(N(f))) \;=\; (\varphi(y_\nu))_\nu\,.$$

□

Zauważmy,że jeśli $f \in \mathbf{R}^n$, $N(f) = (f_1, \ldots, f_n)$, oraz algorytm jest dokładny, $\Phi \equiv \varphi \equiv S$, to numeryczną poprawność algorytmu można równoważnie zapisać jako

$$fl_\nu\,(\Phi(f)) \;=\; (S(f_\nu))_\nu\,.$$

2.4 Rola uwarunkowania zadania

Niech $\Phi(\cdot) = \varphi(N(\cdot))$ będzie algorytmem numerycznie poprawnym dla danych $F_0 \subset F$. Wtedy jego błąd w fl_ν można oszacować następująco:

$$\begin{aligned}
\|S(f) - fl_\nu(\Phi(f))\| \;&=\; \|S(f) - (\varphi(y_\nu))_\nu\| \\
&\le\; \|S(f) - \varphi(y)\| \,+\, \|\varphi(y) - \varphi(y_\nu)\| \,+\, \|\varphi(y_\nu) - (\varphi(y_\nu))_\nu\| \\
&\le\; \|S(f) - \Phi(f)\| \,+\, \|\varphi(y) - \varphi(y_\nu)\| \,+\, K_2\,\nu\,\|\varphi(y_\nu)\| \\
&\le\; \|S(f) - \Phi(f)\| \,+\, (1 + K_2\nu)\|\varphi(y) - \varphi(y_\nu)\| \,+\, K_2\,\nu\,\|\varphi(y)\|,
\end{aligned}$$

przy czym $\|y_\nu - y\| \;\le\; K_1\,\nu\,\|y\|$. Stąd w szczególności wynika, że jeśli algorytm jest numerycznie poprawny i ciągły ze względu na informację y, to

$$\lim_{\nu\to 0} \|S(f) - fl_\nu(\Phi(f))\| \;=\; \|S(f) - \Phi(f)\|.$$

To znaczy, że dla dostatecznie silnej arytmetyki algorytm będzie się zachowywał w fl_ν prawie tak jak w arytmetyce idealnej.

Z powyższych wzorów wynika, że błąd w fl_ν algorytmu numerycznie poprawnego zależy w dużym stopniu od:

- dokładności algorytmu w arytmetyce idealnej,
- dokładności ν arytmetyki fl_ν,
- wrażliwości algorytmu na małe względne zaburzenia informacji y.

Ponieważ dwa pierwsze punkty są raczej oczywiste, poświęcimy trochę więcej uwagi jedynie trzeciemu.

Jeśli φ spełnia warunek Lipschitza ze stałą L, a dokładniej

$$\|\varphi(y_\nu) - \varphi(y)\| \;\le\; L\,\|y_\nu - y\|,$$

to

$$\begin{aligned}
&\|S(f) - fl_\nu(\Phi(f))\| \\
&\quad\le\; \|S(f) - \Phi(f)\| \,+\, (1 + K_2\nu)L\|y_\nu - y\| \,+\, K_2\nu\|\varphi(y)\| \\
&\quad\le\; \|S(f) - \Phi(f)\| \,+\, (1 + K_2\nu)LK_1\nu\|y\| \,+\, K_2\nu\|\varphi(y)\|.
\end{aligned}$$

W tym przypadku błędy zaokrągleń zwiększają błąd bezwzględny algorytmu proporcjonalnie do ν.

Bardziej jednak interesuje nas błąd *względny*. Wybierzmy "małe" $\eta \geq 0$ i przypuśćmy, że

$$\|\varphi(y_\nu) - \varphi(y)\| \;\leq\; M\,K_1\,\nu\,\max(\eta, \|\varphi(y)\|),$$

dla pewnej M niezależnej od y, tzn. błąd względny informscji, $\|y_\nu - y\| \leq K_1\nu\|y\|$, przenosi się na błąd względny wyniku (w arytmetyce idealnej) ze "współczynnikiem wzmocnienia" M, albo na błąd bezwzględny ze współczynnikiem $M\eta$. (Zauważmy, że gdybyśmy wzięli $\eta = 0$ to dla y takiej, że $\varphi(y) = 0$ musiałoby być $\varphi(y_\nu) = 0$, co zwykle, choć nie zawsze, nie jest prawdą.) Wtedy

$$\begin{aligned}
&\|S(f) - fl_\nu(\Phi(f))\| \\
&\quad\leq\; \|S(f) - \Phi(f)\| + (1 + K_2\nu)MK_1\nu\max(\eta, \|\varphi(y)\|) + K_2\nu\|\varphi(y)\| \\
&\quad=\; \|S(f) - \Phi(f)\| \,+\, \nu\Big(MK_1(1 + K_2\nu) + K_2\Big)\max(\eta, \|\varphi(y)\|).
\end{aligned}$$

W szczególności, gdy algorytm jest dokładny i korzysta z pełnej informacji o f, tzn. $S \equiv \Phi \equiv \varphi$, to błąd

$$\frac{\|S(f) - fl_\nu(\Phi(f))\|}{\max(\eta, \|S(f)\|)} \;\leq\; \Big(MK_1(1 + K_2\nu) + K_2\Big)\,\nu \;\approx\; (M\,K_1 \,+\, K_2)\,\nu.$$

Stąd wynika, że jeśli $(MK_1 + K_2)\nu \ll 1$ to błąd względny algorytmu w fl_ν jest mały, o ile $\|S(f)\| \geq \eta$. Błąd względny jest proporcjonalny do dokładności ν, arytmetyki fl_ν, współczynników proporcjonalności K_i algorytmu numerycznie poprawnego, oraz do wrażliwości M zadania S na małe względne zaburzenia danych.

Zwróćmy uwagę na istotny fakt, że interesują nas właściwie tylko te zaburzenia danych (informacji), które powstają przy analizie algorytmu numerycznie poprawnego. I tak, jeśli algorytm jest numerycznie poprawny z pozornymi zaburzeniami danych w normie, to trzeba zbadać wrażliwość zadania ze względu na zaburzenia danych w normie. Jeśli zaś mamy pozorne zaburzenia "po współrzędnych" (co oczywiście implikuje pozorne zaburzenia w normie) to wystarczy zbadać uwarunkowanie zadania ze względu na zaburzenia "po współrzędnych", itd.

Zadania, które nie są zbyt wrażliwe na "małe" względne zaburzenia danych, tzn. dla których M jest "niewielkie", nazywamy ogólnie zadaniami *dobrze uwarunkowanymi*.

2.5 Przykłady

Podamy teraz proste przykłady zadań, które mogą być rozwiązane algorytmami mumerycznie poprawnymi. Oszacujemy też błędy tych algorytmów.

2.5.1 Iloczyn skalarny

Załóżmy. że dla danych ciągów rzeczywistych o ustalonej długości n, a_j, b_j, $1 \leq j \leq n$, chcemy obliczyć

$$S(a,b) \;=\; \sum_{j=1}^{n} a_j b_j.$$

Rozpatrzymy algorytm dokładny zdefiniowany powyższym wzorem i korzystający z pełnej informacji o kolejnych współrzednych.

Oznaczmy przez $\tilde{a}_j$ i $\tilde{b}_j$ reprezentacje liczb a_j i b_j w fl_ν, $\tilde{a}_j = a_j(1+\alpha_j)$, $\tilde{b}_j = b_j(1+\beta_j)$, oraz przez γ_j i δ_j błędy względne powstałe przy kolejnych mnożeniach i dodawaniach. Oczywiście $|\alpha_j|, |\beta_j|, |\gamma_j|, |\delta_j| \leq \nu$. Otrzymujemy

$$\begin{aligned}
fl_\nu\left(\sum_{j=1}^{n} a_j b_j\right) \;&=\; \Big((fl_\nu(\sum_{j=1}^{n-1} a_j b_j) \;+\; \tilde{a}_n \tilde{b}_n (1+\gamma_n) \Big)(1+\delta_n) \;=\; \ldots \\
&=\; \Big(\cdots \Big(\tilde{a}_1 \tilde{b}_1 (1+\gamma_1) + \tilde{a}_2 \tilde{b}_2 (1+\gamma_2) \Big)(1+\delta_2) \\
&\qquad\qquad + \cdots + \tilde{a}_n \tilde{b}_n (1+\gamma_n) \Big)(1+\delta_n) \\
&=\; \tilde{a}_1 \tilde{b}_1 (1+\gamma_1)(1+\delta_2)\cdots(1+\delta_n) \\
&\qquad\qquad + \cdots + \tilde{a}_j \tilde{b}_j (1+\gamma_j)(1+\delta_j)\cdots(1+\delta_n) \\
&=\; \sum_{j=1}^{n} a_j b_j (1+e_j),
\end{aligned}$$

gdzie w przybliżeniu (tzn. gdy $\nu \to 0$) mamy $|e_1| \,\tilde{\leq}\, (n+2)\nu$ i $|e_j| \,\tilde{\leq}\, (n-j+4)\nu$, $2 \leq j \leq n$. Algorytm naturalny jest więc numerycznie poprawny w całym zbiorze danych, gdyż wynik otrzymany w fl_ν można zinterpretować jako dokładny wynik dla danych $a_{\nu,j} = a_j$ i $b_{\nu,j} = b_j(1+e_j)$, przy czym $\|b_\nu - b\|_p \,\tilde{\leq}\, (n+2)\nu \|b\|_p$.

Zobaczmy teraz, jak błąd we współrzędnych b_j wpływa na błąd wyniku. Mamy

$$\begin{aligned}\Big|\sum_{j=1}^n a_jb_j - fl_\nu\Big(\sum_{j=1}^n a_jb_j\Big)\Big| &= \Big|\sum_{j=1}^n a_jb_j - \sum_{j=1}^n a_jb_j(1+e_j)\Big| \\ &= \Big|\sum_{j=1}^n e_ja_jb_j\Big| \le \sum_{j=1}^n |e_j||a_jb_j| \\ &\tilde{\le} (n+2)\nu\sum_{j=1}^n |a_jb_j|.\end{aligned}$$

Stąd dla $\eta \ge 0$

$$\frac{|\sum_{j=1}^n a_jb_j - fl_\nu(\sum_{j=1}^n a_jb_j)|}{\max(\eta, |\sum_{j=1}^n a_jb_j|)} \tilde{\le} K_\eta\,(n+2)\,\nu,$$

gdzie

$$K_\eta = K_\eta(a,b) = \frac{\sum_{j=1}^n |a_jb_j|}{\max(\eta, |\sum_{j=1}^n a_jb_j|)}.$$

Zauważmy, że jeśli iloczyny a_jb_j są wszystkie dodatnie albo wszystkie ujemne, to $K_\eta = 1$, tzn. zadanie jest dobrze uwarunkowane, a błąd względny jest zawsze na poziomie co najwyżej $n\nu$. W tym przypadku algorytm zachowuje się bardzo dobrze, o ile liczba n składników nie jest horendalnie duża. W ogólności jednak K_η może być liczbą dowolnie dużą i wtedy nie możemy być pewni uzyskania dobrego wyniku w fl_ν.

2.5.2 Całkowanie

Zadanie całkowania z Przykładu 1.3 często rozwiązuje się (a raczej przybliża) stosując formułę

$$\Phi(f) = \varphi(y) = \sum_{j=1}^n c_jy_j,$$

gdzie informacja $y_j = f(t_j)$. t_j są tu ustalonymi punktami z $[a,b]$, a c_j ustalonymi współczynnikami rzeczywistymi, niezależnymi od f. Algo-

rytmy korzystające z takich formuł nazywamy *kwadraturami*, a przykładem jest zwykła suma Riemanna,

$$\Phi_R(f) \;=\; \frac{1}{n}\sum_{j=1}^{n} f(t_j^*),$$

gdzie t_j^* należy do przedziału $[a + (j-1)(b-a)/n, a + j(b-a)/n]$, $1 \le j \le n$.

Z analizy algorytmu obliczającego iloczyn skalarny wynika, że kwadratura jest algorytmem numerycznie poprawnym. Mamy bowiem

$$fl_\nu\left(\sum_{j=1}^{n} c_j f(t_j)\right) \;=\; \sum_{j=1}^{n} c_j y_j (1+e_j),$$

gdzie $|e_j| \,\tilde{\le}\, (n+1)\nu$. (W porównaniu z iloczynem skalarnym mamy tu $n+1$ zamiast $n+2$, bo c_j nie jest daną i nie podlega zaburzeniu.) Stąd błąd bezwzględny kwadratury w fl_ν można oszacować następująco:

$$\begin{aligned} &\Big|\int_a^b f(x)\,dx \;-\; fl_\nu\Big(\sum_{j=1}^{n} c_j f(t_j)\Big)\Big| \\ &\quad\le\; \Big|\int_a^b f(x)\,dx \;-\; \sum_{j=1}^{n} c_j f(t_j)\Big| \;+\; (n+1)\,\nu\,\sum_{j=1}^{n} |c_j f(t_j)|. \end{aligned}$$

Uwagi i uzupełnienia

U. 2.1 Wprowadzimy teraz formalną definicję zastosowanej już wcześniej przybliżonej nierówności $\tilde{\le}$. Dla dwóch funkcji piszemy

$$|h_1(\nu)| \;\tilde{\le}\; |h_2(\nu)|$$

gdy

$$\limsup_{\nu\to 0} |h_1(\nu)|/|h_2(\nu)| \;\le\; 1.$$

Na przykład, jeśli $|h(\nu)| \le \sum_{j=1}^{n} K_j\nu^j$ $(K_1 > 0)$, to $|h(\nu)| \,\tilde{\le}\, K_1\nu$, co już zauważyliśmy w przykładzie z iloczynem skalarnym. Zapis ten wyraża prosty fakt, że dla praktycznych wartości ν $(\nu \le 10^{-8})$ wyrazy typu ν^2, ν^3, itd. są tak małe w porównaniu z ν, że można je zaniedbać.

Łatwo sprawdzić, że przy notacji $\tilde{\leq}$ zachodzą następujące fakty. Jeśli $|\varepsilon| \,\tilde{\leq}\, K\nu$, $|\varepsilon_1| \,\tilde{\leq}\, K_1\nu$ i $|\varepsilon_2| \,\tilde{\leq}\, K_2\nu$ (gdzie ε, ε_1, ε_2 są funkcjami ν), to

$$(1+\varepsilon_1)(1+\varepsilon_2) \;=\; (1+\delta), \qquad \text{gdzie} \qquad |\delta| \,\tilde{\leq}\, (K_1+K_2)\nu, \tag{2.3}$$

$$(1+\varepsilon)^{-1} \;=\; (1+\delta), \qquad \text{gdzie} \qquad |\delta| \,\tilde{\leq}\, K\nu, \tag{2.4}$$

$$(1+\varepsilon)^{1/2} \;=\; (1+\delta), \qquad \text{gdzie} \qquad |\delta| \,\tilde{\leq}\, \frac{1}{2}K\nu, \tag{2.5}$$

i ogólnie

$$(1+\varepsilon)^{p} \;=\; (1+\delta), \qquad \text{gdzie} \qquad |\delta| \,\tilde{\leq}\, |p|\nu. \tag{2.6}$$

U. 2.2 Rozpatrzymy teraz zadanie obliczenia wszystkich pierwiastków rzeczywistych równania kwadratowego z Przykładu 1.1. Będziemy zakładać, że model obliczeniowy dopuszcza obliczanie pierwiastków kwadratowych z liczb nieujemnych oraz $fl_\nu(\sqrt{x}) = rd(\sqrt{rd(x)})$.

Okazuje się, że nie umiemy pokazać numerycznej poprawności "szkolnego" algorytmu obliczającego pierwiastki równania bezpośrednio ze wzorów (1.1). Można jednak pokazać numeryczną poprawność drobnej jego modyfikacji wykorzystującej wzory Viete'a.

```
Δ := p ∗ p − q;
if  (Δ = 0)  then  OUT(p)  else
if  (Δ > 0)  then
begin
    Δ1 := sqrt(d);
    if  (p ≥ 0)  then
    begin
        x1 := p + Δ1;
        x2 := q/x1;
    end  else
    begin
        x2 := p − Δ1;
        x1 := q/x2;
    end;
    OUT(x1);   OUT(x2)
end.
```

Mamy bowiem

$$fl_\nu(\Delta(p,q)) \;=\; \Big(p^2(1+\alpha)^2(1+\varepsilon_1) - q(1+\beta)\Big)(1+\varepsilon_2)$$

$$
\begin{aligned}
&= \left(p^2 - q\frac{(1+\beta)}{(1+\alpha)^2(1+\varepsilon_1)}\right)(1+\varepsilon_2)(1+\alpha)^2(1+\varepsilon_1) \\
&= \Big(p^2 - q(1+\delta)\Big)(1+\gamma) \;=\; \Delta(p, q(1+\delta))(1+\gamma),
\end{aligned}
$$

gdzie $|\delta|, |\gamma| \,\tilde{\leq}\, 4\nu$. Wyróżnik obliczony w fl_ν jest więc nieco zaburzonym wyróżnikiem dokładnym dla danych p i $q_\nu = q(1+\delta)$. W szczególności

$$\mathrm{sgn}(fl_\nu(\Delta(p,q))) = \mathrm{sgn}(\Delta(p, q_\nu)).$$

Jeśli $p \geq 0$ to

$$
\begin{aligned}
fl_\nu(x1(p,q)) &= \Big(p(1+\alpha) + \sqrt{fl_\nu(\Delta(p,q))}(1+\varepsilon_3)\Big)(1+\varepsilon_4) \\
&= \Big(p(1+\alpha) + \sqrt{\Delta(p,q_\nu)(1+\gamma)}(1+\varepsilon_3)\Big)(1+\varepsilon_4) \\
&= \left(p + \sqrt{\Delta(p,q_\nu)}\frac{\sqrt{1+\gamma}(1+\varepsilon_3)}{1+\alpha}\right)(1+\varepsilon_4)(1+\alpha) \\
&= \left(p + \sqrt{\Delta(p,q_\nu)}\right)(1+e_1),
\end{aligned}
$$

gdzie $|e_1| \,\tilde{\leq}\, 6\nu$. Zauważmy, że ostatnia równość zachodzi dlatego, że dodajemy liczby tego samego znaku. (Inaczej $|e_1|$ mogłaby być dowolnie duża i tak byłoby w algorytmie szkolnym.) Dla drugiego pierwiastka mamy

$$fl_\nu(x2(p,q)) \;=\; \frac{q(1+\beta)}{fl_\nu(x1(p,q))}(1+\varepsilon_5) \;=\; \frac{q_\nu}{fl_\nu(x1(p,q))}(1+e_2),$$

gdzie $|e_2| \leq 8\nu$.

Podobny wynik otrzymalibyśmy dla $p < 0$. Algorytm zmodyfikowany jest więc numerycznie poprawny, gdyż otrzymane w fl_ν pierwiastki są nieco zaburzonymi dokładnymi pierwiatkami dla danych $p_\nu = p$ i $q_\nu = q(1+\delta)$.

Aby oszacować błąd algorytmu, wystarczy zbadać uwarunkowanie zadania ze względu na zaburzenie danej q, ponieważ pokazaliśmy, że zaburzenia p można przenieść na zaburzenia q i wyniku. Niestety, choć algorytm jest numerycznie poprawny, zaburzenia q mogą sprawić, że nawet znak wyróżnika Δ może być obliczony nieprawidłowo. Na przykład dla $p = 1$ i $q = 1 \pm 10^{t+1}$ mamy $\Delta(p,q) = \mp 10^{t+1}$, ale $\Delta(rd(p), rd(q)) = \Delta(1,1) = 0$. Ogólnie

$$|fl_\nu(\Delta(p,q)) - \Delta(p,q)| \;\tilde{\leq}\; 4\nu(p^2 + 2|q|),$$

a więc tylko dla $|\Delta(p,q)| = |p^2 - q| > 4\nu(p^2 + 2|q|)$ możemy być pewni obliczenia właściwego znaku Δ. Przy tym warunku oraz $\Delta > 0$ błąd danych przenosi się w normie euklidesowej na błąd wyniku następująco:

$$\begin{aligned}
&\Big((x1(p,q) - x1(p,q_\nu))^2 + (x2(p,q) - x2(p,q_\nu))^2\Big)^{1/2} \\
&= \frac{\sqrt{2}|\delta q|}{\sqrt{p^2-q} + \sqrt{p^2-q_\nu}} \;\tilde{\le}\; 2\sqrt{2}\nu\frac{|q|}{\sqrt{p^2-q}} \\
&= 2\sqrt{2}\nu\frac{|q|/p^2}{\sqrt{1-q/p^2}\max(\eta/|p|, \sqrt{2(1+(1-q/p^2))})} \\
&\qquad\cdot \max(\eta, (x1(p,q)^2 + x2(p,q)^2)^{1/2}).
\end{aligned}$$

Stąd widać, że zadanie jest dobrze uwarunkowane dla $q/p^2 \ll 1$ i może być źle uwarunkowane dla $q/p^2 \approx 1$. W ostatnim przypadku nie możemy być pewni otrzymania dobrego wyniku w fl_ν.

Ćwiczenia

Ćw. 2.1 Pokazać równości (2.3)–(2.6).

Ćw. 2.2 Niech $0 < a_1 < a_2 < \cdots a_n$. Czy z punktu widzenia błędów w fl_ν lepiej jest policzyć sumę tych liczb w kolejności od najmniejszej do największej czy odwrotnie?

Ćw. 2.3 Aby obliczyć $S(a,b) = a^2 - b^2$ można zastosować dwa algorytmy: $\Phi_1(a,b) = a*a - b*b$ oraz $\Phi_2(a,b) = (a+b)*(a-b)$. Pokazać, że oba algorytmy są numerycznie poprawne, ale drugi z nich wywołuje mniejszy błąd względny wyniku w przypadku, gdy $rd(a) = a$ i $rd(b) = b$.

Ćw. 2.4 Pokazać, że naturalny algorytm obliczania cosinusa kąta między dwoma wektorami $\vec{a}, \vec{b} \in \mathbf{R}^n$,

$$\cos(a,b) = \frac{\sum_{j=1}^n a_j b_j}{\sqrt{\left(\sum_{j=1}^n a_j^2\right)\left(\sum_{j=1}^n b_j^2\right)}},$$

jest numerycznie poprawny. Oszacować błąd względny wyniku w fl_ν.

Ćw. 2.5 Pokazać, że naturalny algorytm obliczania $\|A\vec{x}\|_2$ dla danej macierzy $A \in \mathbf{R}^{n\times n}$ i wektora $\vec{x} \in \mathbf{R}^n$ jest numerycznie poprawny. Dokładniej,

$$fl_\nu(\|A\vec{x}\|_2) \;=\; (A+E)\vec{x},$$

gdzie $\|E\|_2 \,\tilde{\le}\, 2(n+2)\sqrt{n}\nu\|A\|_2$. Ponadto, jeśli A jest nieosobliwa to

$$|fl_\nu(\|A\vec{x}\|_2) - \|A\vec{x}\|_2| \;\tilde{\le}\; 2(n+2)\sqrt{n}\,\nu\,\left(\|A\|_2\|A^{-1}\|_2\right)\,\|A\vec{x}\|_2.$$

Ćw. 2.6 Niech Φ będzie algorytmem numerycznie poprawnym w zbiorze danych $f \in F_0$, przy czym dla małych ν, $fl_\nu(\Phi(f)) = \varphi(y_\nu)$, gdzie $\|y_\nu - y\| \le K\nu\|y\|$ i K nie zależy od ν i f $(y = N(f))$. Pokazać, że w ogólności Φ nie musi być "numerycznie poprawny po współrzędnych", tzn. w ogólności nie istnieje bezwzględna stała K_1 taka, że dla małych ν i dla dowolnej $f \in F_0$

$$|y_{\nu,j} - y_j| \;\le\; K_1\,\nu\,|y_j|, \qquad 1 \le j \le n,$$

gdzie $y = (y_1, \ldots, y_n)$.

æ ææ

Rozdział 3

Układy równań liniowych

Rozpoczynamy analizę metod dokładnych rozwiązywania układów równań liniowych postaci

$$A\vec{x} \;=\; \vec{b}, \tag{3.1}$$

gdzie $A = (a_{i,j})_{i,j=1}^n$ jest macierzą nieosobliwą ($\det A \neq 0$) wymiaru $n \times n$, a $\vec{b} = (b_i)_{i=1}^n$ jest wektorem z $\mathbf{R}^n$. Zakładamy, że informacja o zadaniu dana jest przez współczynniki $a_{i,j}$ macierzy i współrzędne b_i wektora. W wyniku powinniśmy uzyskać współrzędne x_j^* wektora rozwiązania $\vec{x}^* = (x_j^*)_{j=1}^n$,

$$\vec{x}^* \;=\; A^{-1}\vec{b}.$$

Przypomnijmy, że nieosobliwość macierzy A zapewnia, że rozwiązanie $\vec{x}^*$ istnieje i jest wyznaczone jednoznacznie.

3.1 Układy z macierzą trójkątną

Szczególnym przypadkiem (3.1) jest układ z macierzą trójkątną A. Będą nas szczególnie interesować macierze *trójkątne górne*, dla których $a_{i,j} = 0$ gdy $i > j$, oraz macierze *trójkątne dolne* z jedynkami na przekątnej, tzn. $a_{i,j} = 0$, $i < j$, oraz $a_{i,i} = 1$. Macierze pierwszego rodzaju będziemy oznaczać przez R, a drugiego rodzaju przez L.

Układ z macierzą trójkątną górną

$$R\vec{x} \;=\; \vec{c}, \tag{3.2}$$

$R = (r_{i,j})$, $\vec{c} = (c_j)$, można rozwiązać stosując algorytm:

$x_n^* := c_n / r_{n,n};$
for $i := n-1$ **downto** 1 **do**
$x_i^* := \left(c_i - \sum_{j=i+1}^{n} r_{i,j} x_j^*\right) / r_{i,i}.$

(Algorytm ten jest wykonalny, ponieważ nieosobliwość macierzy implikuje, że $r_{i,i} \neq 0$, $\forall i$.) Podobnie, układ $L\vec{x} = \vec{c}$ rozwiązujemy algorytmem:

$x_1^* := c_1;$
for $i := 2$ **to** n **do**
$x_i^* := c_i - \sum_{j=1}^{i-1} l_{i,j} x_j^*.$

Oba algorytmy wymagają rzędu $n^2/2$ mnożeń lub dzieleń i $n^2/2$ dodawań lub odejmowań, a więc rzędu n^2 wszystkich działań arytmetycznych.

3.2 Eliminacja Gaussa

Eliminacja Gaussa jest algorytmem dokładnym rozwiązywania układów równań z macierzą pełną A, polegającym na sprowadzeniu układu wyjściowego (3.1) do układu trójkątnego (3.2).

W postaci rozwiniętej układ (3.1) ma postać

$$\begin{array}{lllllllll}
a_{1,1}x_1 & + & a_{1,2}x_2 & + & \cdots & + & a_{1,n}x_n & = & b_1 \\
a_{2,1}x_1 & + & a_{2,2}x_2 & + & \cdots & + & a_{2,n}x_n & = & b_2 \\
\ldots & & & & & & & & \\
a_{n,1}x_1 & + & a_{n,2}x_2 & + & \cdots & + & a_{n,n}x_n & = & b_n.
\end{array}$$

Zauważmy, że jeśli od i-tego wiersza tego układu odejmiemy wiersz pierwszy pomnożony przez $l_{i,1} = a_{i,1}/a_{1,1}$, dla $i = 2, 3, \ldots, n$ (zakładamy, że $a_{1,1} \neq 0$) to otrzymamy układ równoważny $A^{(1)}\vec{x} = \vec{b}^{(1)}$,

$$\begin{array}{lllllllll}
a_{1,1}x_1 & + & a_{1,2}x_2 & + & \cdots & + & a_{1,n}x_n & = & b_1 \\
 & & a_{2,2}^{(1)}x_2 & + & \cdots & + & a_{2,n}^{(1)}x_n & = & b_2^{(1)} \\
 & & \ldots & & & & & & \\
 & & a_{n,2}^{(1)}x_2 & + & \cdots & + & a_{n,n}^{(1)}x_n & = & b_n^{(1)},
\end{array}$$

z wyrugowaną zmienną x_1 z równań od drugiego do n-tego. Teraz z kolei możemy od i-tego wiersza odjąć wiersz drugi pomnożony przez $l_{i,2} = a_{i,2}^{(1)}/a_{2,2}^{(1)}$, dla $i = 3, 4, \ldots, n$ (o ile $a_{2,2}^{(1)} \neq 0$), aby otrzymać układ $A^{(2)}\vec{x} = \vec{b}^{(2)}$ z wyrugowanymi zmiennymi x_1 i x_2 z równań od trzeciego do n-tego. Postępując tak dalej otrzymujemy po $n-1$ krokach układ $A^{(n-1)}\vec{x} = \vec{b}^{(n-1)}$ postaci

$$\begin{array}{ccccccccc} a_{1,1}^{(0)}x_1 & + & a_{1,2}^{(0)}x_2 & + & \cdots & + & a_{1,n}^{(0)}x_n & = & b_1 \\ & & a_{2,2}^{(1)}x_2 & + & \cdots & + & a_{2,n}^{(1)}x_n & = & b_2^{(1)} \\ & & & & \ddots & & & & \\ & & & & & & a_{n,n}^{(n-1)}x_n & = & b_n^{(n-1)}. \end{array}$$

Układ ten jest trójkątny górny, $R\vec{x} = \vec{c}$, z macierzą $R = A^{(n-1)}$ i wektorem $\vec{c} = \vec{b}^{(n-1)}$, można go więc rozwiązać znaną już metodą.

Kolejne operacje w eliminacji Gaussa można zapisać następująco:

```
for k := 1 to n-1 do
for i := k+1 to n do
begin
    l_{i,k} := a_{i,k}^{(k-1)} / a_{k,k}^{(k-1)};
    b_i^{(k)} := b_i^{(k-1)} - l_{i,k} * b_k^{(k-1)};
    for j := k+1 to n do
        a_{i,j}^{(k)} := a_{i,j}^{(k-1)} - l_{i,k} * a_{k,j}^{(k-1)}
end.
```

Zauważmy, że k-ty krok wymaga rzędu $(n-k)^2$ dodawań lub odejmowań i tyle samo mnożeń lub dzieleń. Koszt całej eliminacji wynosi więc $2\sum_{k=1}^{n-1}(n-k)^2 \approx (2/3)n^3$ operacji arytmetycznych. Ponieważ układ trójkątny potrafimy rozwiązać kosztem n^2, całkowity koszt rozwiązania układu $A\vec{x} = \vec{b}$ jest również rzędu $(2/3)n^3$.

Opisany powyżej proces eliminacyjny załamie się gdy dla pewnego k element $a_{k,k}^{(k-1)}$ będzie zerem. W takim przypadku prosta modyfikacja polegająca na przestawieniu wiersza k-tego układu z wierszem s_k-tym $(k+1 \leq s_k \leq n)$ dla którego $a_{s_k,k}^{(k-1)} \neq 0$, pozwala kontynuować eliminację. Taki niezerowy element oczywiście istnieje, bo inaczej macierz $A^{(k-1)}$, a tym samym i macierz A, byłaby osobliwa.

Najczęściej wybieramy s_k tak, aby

$$|a_{s_k,k}^{(k-1)}| \;=\; \max_{k+1\le i\le n} |a_{i,k}^{(k-1)}|,$$

co wymaga dodatkowo rzędu $n^2/2$ porównań liczb rzeczywistych i żadnych operacji arytmetycznych. Taki wybór elementu niezerowego prowadzi do *eliminacji Gaussa z wyborem elementu głównego w kolumnie.* Wtedy współczynniki $l_{i,k} = a_{i,k}^{(k-1)}/a_{k,k}^{(k-1)}$ są wszystkie co do modułu nie większe od jedności co, jak się przekonamy później, ma niebagatelne znaczenie dla własności numerycznych całego algorytmu.

3.3 Rozkład trójkątny macierzy

Podamy teraz ważne twierdzenia wiążące eliminację Gaussa z rozkładem trójkątno-trójkątnym macierzy A.

Twierdzenie 3.1 *Eliminacja Gaussa bez przestawień wierszy (o ile jest wykonalna) jest równoważna rozkładowi macierzy A na iloczyn macierzy trójkątnej dolnej $L = (l_{i,j})$ (z jedynkami na przekątnej) i trójkątnej górnej $R = (r_{i,j})$. Dokładniej,*

$$A \;=\; L \cdot R,$$

gdzie

$$l_{i,j} \;=\; \begin{cases} a_{i,j}^{(i-1)}/a_{i,i}^{(i-1)} & i > j, \\ 1 & i = j, \\ 0 & i < j, \end{cases}$$

oraz

$$r_{i,j} \;=\; \begin{cases} a_{i,j}^{(i-1)} & i < j, \\ 0 & i \ge j. \end{cases}$$

Dowód Zauważmy, że rugowanie elementów pod główną przekątną w k-tej kolumnie ($k = 1, 2, \ldots, n-1$) odpowiada mnożeniu układu

$A^{(k-1)}\vec{x} = \vec{b}^{(k-1)}$ z lewej strony przez macierz

$$L_k = \begin{pmatrix} 1 & & & & & & \\ & 1 & & & & & \\ & & \ddots & & & & \\ & & & 1 & & & \\ & & & -l_{k+1,k} & 1 & & \\ & & & \vdots & & \ddots & \\ & & & -l_{n,k} & & & 1 \end{pmatrix}.$$

Stąd

$$L_{n-1}L_{n-2}\cdots L_2L_1A = R$$

i

$$A = (L_1^{-1}L_2^{-1}\cdots L_{n-2}^{-1}L_{n-1}^{-1})\,R.$$

Łatwo sprawdzić, że macierz L_k^{-1} różni się od L_k tylko tym, że elementy $-l_{i,k}$ dla $k+1 \le i \le n$ zmieniają się na $l_{i,k}$. Ponadto, mnożenie macierzy L_k^{-1} przez siebie w podanej kolejności odpowiada "przepisywaniu" elementów $l_{i,k}$ do macierzy wynikowej tak, że ostatecznie mamy $L_1^{-1}\cdots L_{n-1}^{-1} = L$ i $A = LR$. $\square$

Zobaczmy teraz jak wybór elementu głównego w kolumnie i przestawianie odpowiednich wierszy macierzy układu wpływa na rozkład macierzy A z Twierdzenia 3.1.

Przestawienie wiersza k-tego z s-tym w danej macierzy jest formalnie równoważne pomnożeniu tej macierzy z lewej strony przez macierz *permutacji elementarnej* $P_{k,s}$. Jest to macierz, która powstaje z macierzy jednostkowej przez przestawienie jej k-tego wiersza z s-tym (albo k-tej kolumny z s-tą). Eliminacja Gaussa z wyborem elementu głównego w kolumnie odpowiada więc mnożeniu macierzy A z lewej strony kolejno przez macierze permutacji elementarnej P_{k,s_k} i macierze "eliminacji" L_k tak, że

$$L_{n-1}P_{n-1,s_{n-1}}L_{n-2}P_{n-2,s_{n-2}}\cdots L_2P_{2,s_2}L_1P_{1,s_1}A = R.$$

Ponieważ $P_{k,l}^2 = I$ (identyczność), mamy

$$P_{2,s_2}L_1P_{1,s_1}A = (P_{2,s_2}L_1P_{2,s_2})(P_{2,s_2}P_{1,s_1}A) = L_1^{(1)}(P_{2,s_2}P_{1,s_1}A),$$

gdzie $L_1^{(1)}$ różni się od L_1 tym, że zostały przestawione elementy $(2,1)$ i $(s_2,1)$. Następnie,

$$\begin{aligned} & P_{3,s_3} L_2 L_1^{(1)} (P_{2,s_2} P_{1,s_1} A) \\ &= (P_{3,s_3} L_2 P_{3,s_3})(P_{3,s_3} L_1^{(1)} P_{3,s_3})(P_{3,s_3} P_{2,s_2} P_{1,s_1} A) \\ &= L_2^{(2)} L_1^{(2)} (P_{3,s_3} P_{2,s_2} P_{1,s_1} A), \end{aligned}$$

gdzie $L_1^{(2)}$ różni się od $L_1^{(1)}$ tym, że zostały przestawione elementy $(3,1)$ i $(s_3,1)$, a $L_2^{(2)}$ różni się od L_2 przestawieniem elementów $(3,2)$ i $(s_3,2)$. Postępując tak dalej otrzymujemy

$$\begin{aligned} & L_{n-1} P_{n-1,s_{n-1}} \cdots L_2 P_{2,s_2} L_1 P_{1,s_1} A \\ &= L_{n-1}^{(n-1)} L_{n-1}^{(n-1)} \cdots L_2^{(n-1)} L_1^{(n-1)} (P_{n-1,s_{n-1}} \cdots P_{2,s_2} P_{1,s_1}) A = R, \end{aligned}$$

gdzie $L_k^{(n-1)}$ różni się od L_k jedynie pewną permutacją elementów w k-tej kolumnie pod główną przekątną,

$$L_k^{(n-1)} = P_{n-1,s_{n-1}} \cdots P_{k+1,s_{k+1}} L_k P_{k+1,s_{k+1}} \cdots P_{n-1,s_{n-1}}.$$

Postępując dalej tak, jak w przypadku eliminacji bez przestawień wierszy, otrzymujemy następujący wniosek.

Wniosek 3.1 *Eliminacja Gaussa z wyborem elementu głównego w kolumnie jest wykonalna i jest równoważna rozkładowi*

$$P \cdot A = L \cdot R,$$

gdzie $P = P_{n-1,s_{n-1}} \cdots P_{2,s_2} P_{1,s_1}$ *jest macierzą permutacji, a* L *i* R *są macierzami konstruowanymi tak jak w Twierdzeniu 3.1, ale dla macierzy* PA *zamiast* A*. Ponadto wszystkie elementy macierzy* L *są co do modułu nie większe od jedności.* □

Dodajmy, że znalezienie czynników rozkładu $PA = LR$ dla danej macierzy A kosztuje tyle samo co eliminacja Gaussa z wyborem elementu głównego, czyli $(2/3)n^3$ operacji arytmetycznych i $n^2/2$ porównań.

W praktycznej realizacji rozkładu $PA = LR$ oczywiście nie musimy przy przestawieniu wierszy dokonywać fizycznie przepisań elementów

danego wiersza do innego. Wystarczy mieć dodatkowy n-wymiarowy wektor permutacji $p[]$ interpretowany w ten sposób, że $p[i]$ jest wskaźnikiem do i-tego wiersza macierzy A. Przestawienie wierszy możemy wtedy realizować po prostu przez zamianę wskaźników, czyli elementów wektora p. (Zob. U. 3.5.)

Twierdzenie 3.1 i Wniosek 3.1 mają nie tylko teoretyczne znaczenie. Rozkładu macierzy na czynniki trójkątne (co w praktyce sprowadza się do pamiętania kolejnych permutacji i mnożników $l_{i,j}$) opłaca się dokonywać w przypadku, gdy zadanie polega na rozwiązaniu nie jednego, ale wielu układów równań z tą samą macierzą A i ze zmieniającym się wektorem prawej strony,

$$A\,\vec{x} \;=\; \vec{b}_s, \qquad \text{dla} \quad 1 \le s \le k.$$

Jeśli każdy z tych układów rozwiązujemy "od początku" to musimy wykonać $(2/3)n^3k$ operacji arytmetycznych. Koszt ten możemy znacznie zredukować, jeśli dokonamy wstępnego rozkładu $PA = LR$. Wtedy układ $A\vec{x} = \vec{b}_s$ jest równoważny układowi $LR\vec{x} = P\vec{b}_s$, a więc jego rozwiązanie sprowadza się do rozwiązania dwóch znanych już układów trójkątnych:

$$L\,\vec{y} \;=\; P\,\vec{b}_s$$

i

$$R\,\vec{x} \;=\; \vec{y}.$$

Taki sposób postępowania wymaga tylko $(2/3)n^3 + 2n^2k = 2n^2(n/3 + k)$ operacji arytmetycznych, co np. przy $k = n$ powoduje obniżenie rzędu kosztu z n^4 do n^3.

Uwagi i uzupełnienia

U. 3.1 Zaproponowany algorytm dla układów $R\vec{x} = \vec{c}$ z macierzą trójkątną R ma prawie optymalną złożoność kombinatoryczną (n^2) wśród algorytmów dokładnych. Zauważmy bowiem, że jeśli algorytm jest dokładny to wykorzystuje wszystkie dane $r_{i,j}$ i c_j, $1 \le i, j \le n$ (zob. Ćw.3.1). Ponieważ każda kolejna operacja arytmetyczna $(+, -, *, /)$ może wykorzystywać co najwyżej dwie nowe dane, a wszystkich danych jest $n(n+1)/2 + n \approx n^2/2$, algorytm dokładny musi wykonywać co najmniej $n^2/4$ działań.

U. 3.2 Znane są algorytmy dokładne, które rozwiązują układ równań z macierzą pełną kosztem proporcjonalnym do n^p, gdzie $p < 3$, a więc mniejszym niż w eliminacji Gaussa. Algorytmy te są jednak nieużyteczne ze względu na skomplikowaną konstrukcję, zwykle dużą stłą przy n^p, a także ze względu na złe własności numeryczne. Oczywiście, ograniczeniem dolnym na wykładnik p w koszcie rozwiązania układu z macierzą pełną jest 2 (bo mamy rzędu n^2 danych), ale nie wiadomo, czy może on być osiągnięty przez jakiś algorytm.

U. 3.3 Układy równań z tą samą macierzą, ale ze zmieniającą się prawą stroną równania powstają często przy rozwiązywaniu, np. równań różniczkowych cząstkowych, gdzie prawa strona układu odpowiada zmieniającym się warunkom brzegowym. Z wieloma układami tego typu mamy również do czynienia gdy chcemy znaleźć macierz odwrotną do danej macierzy $A \in \mathbf{R}^{n\times n}$. Rzeczywiście, kolejne kolumny macierzy A^{-1} są rozwiązaniami układów

$$A\,\vec{x} \;=\; \vec{e}_j, \qquad 1 \le j \le n,$$

gdzie $\vec{e}_j$ oznacza j-ty wersor.

U. 3.4 Obok wyboru elementu głównego w kolumnie dokonuje się również wyboru elementu gównego w całej macierzy. To znaczy, w k-tym kroku eliminacyjnym wybiera się element $a^{(k-1)}_{s_k,t_k}$, $k \le s_k, t_k \le n$, taki, że

$$|a^{(k-1)}_{s_k,t_k}| \;=\; \max_{k\le i,j\le n} |a^{(k-1)}_{i,j}|,$$

a następnie przestawia się wiersze k-ty z s_k-tym i kolumny k-tą z t_k-tą w macierzy $A^{(k-1)}$. Zauważmy, że przestawienie kolumn macierzy odpowiada mnożeniu jej z prawej strony przez macierz permutacji P_{k,t_k}. Przyjmując $\overline{P} = P_{1,t_1}P_{2,t_2}\cdots P_{n-1,t_{n-1}}$ i korzystając z Twierdzenia 3.1 i Wniosku 3.1 mamy, że eliminacja Gaussa z wyborem elementu głównego w całej macierzy jest równoważna rozkładowi

$$P\,A\,\overline{P} \;=\; L\,R.$$

U. 3.5 Podamy teraz jedną z możliwych implementacji algorytmu rozkładu macierzy na iloczyn macierzy trójkątnych, $PA = LR$. W poniższym programie współczynniki macierzy wyjściowej są pamiętane w tablicy $a[,]$. Po wykonaniu programu, informacja o permutacji wierszy zapamiętana będzie w wektorze $p[]$. W $a[p[i],j]$ będą elementy (i,j) macierzy L dla $i > j$, oraz macierzy R dla $i \le j$.

```
{ inicjacja wskaźników }
for  i := 1  to  n  do  p[i] := i;
for  j := 1  to  n − 1  do
begin
    { wybór elementu głównego }
    im := j;   val := abs(a[p[j], j]);
    for  i := j + 1  to  n  do
    begin
        v := abs(a[p[i], j]);
        if  v > val  then
        begin
            im := i;   val := i
        end;
        { zamiana wskaźników }
        s := p[im];   p[im] := p[j];   p[j] := s;
        { eliminacja }
        v := a[s, j];
        for  i := j + 1  to  n  do
        begin
            im := p[i];
            l := a[im, j]/v;
            a[im, j] := l;
            for  k := j + 1  to  n  do
            a[im, k] := a[im, k] − l ∗ a[s, k]
        end;
    end;
end.
```

U. 3.6 Przedstawiony algorytm eliminacji Gaussa rozwiązuje układy z macierzami dowolnej postaci. Jeśli macierz A jest szczególnej postaci to czasem proste jego modyfikacje pozwalają znacznie zmniejszyć koszt uzyskania rozwiązania. Na przykład, niech A będzie macierzą trójdiagonalną postaci

$$A = \begin{pmatrix} a_1 & c_1 & & & & \\ b_2 & a_2 & c_2 & & & \\ & b_3 & a_3 & c_3 & & \\ & & \ddots & \ddots & \ddots & \\ & & & & b_n & a_n \end{pmatrix}.$$

Wtedy układ $A\vec{x} = \vec{e}$ można rozwiązać stosując tzw. algorytm "przegania-nia":

$d_1 := a_1; \quad f_1 := e_1;$
for $i := 2$ **to** n **do**
begin
$\quad l := b_i / a_{i-1};$
$\quad d_i := a_i - l * c_{i-1};$
$\quad f_i := e_i - l * f_{i-1}$
end;
$x_1^* := f_n;$
for $i := n-1$ **downto** 1 **do**
$x_i^* := f_i - c_i * x_{i+1}^*.$

Okazuje się, że jeśli macierz A ma dominującą przekątną, tzn.

$$|a_i| \geq |b_i| + |c_i|, \qquad 1 \leq i \leq n, \tag{3.3}$$

$(b_1 = 0 = c_n)$ i przynajmniej dla jednego i mamy powyżej nierówność $<$, to algorytm przeganiania jest wykonalny bez przestawień wierszy. Ponadto wymaga on $7n$ operacji arytmetycznych, a więc jest prawie optymalny.

U. 3.7 Innym ważnym przykładem macierzy szczególnej postaci są macierze symetryczne i dodatnio określone. Są to macierze spełniające $A = A^T$ oraz

$$\vec{x}^T A \vec{x} > 0, \qquad \forall \vec{x} \neq 0.$$

Dla takich macierzy można nieco zmniejszyć koszt kombinatoryczny i zużycie pamięci przeprowadzając eliminację tak, aby otrzymać rozkład

$$A = L \cdot D \cdot L^T$$

zamiast $PA = LR$, przy czym L jest tu jak zwykle macierzą trójkątną dolną z jedynkami na przekątnej, a D jest macierzą diagonalną z dodatnimi elementami na diagonali.

Rozkład taki przeprowadzamy mnożąc macierz A przez znane już macierze eliminacji nie tylko z lewej, ale też i z prawej strony, bez przestawień wierszy. Bez zmniejszenia ogólności rozpatrzymy tylko pierwszy krok. W tym celu, zauważmy najpierw, że $a_{1,1} = \vec{e}_1^T A \vec{e}_1 > 0$ (gdzie $\vec{e}_1$ jest pierwszym wersorem), a więc nie musimy przestawiać wierszy, bo element na diagonali jest niezerowy. W pierwszym kroku mnożymy macierz A z lewej strony przez odpowiednią macierz L_1, a potem z prawej przez L_1^T. Kluczem do zrozumienia algorytmu jest uwaga, że efektem mnożenia macierzy $L_1 A$

z prawej strony przez L_1^T jest wyzerowanie elementów pierwszego wiersza poza $a_{1,1}$ i pozostawienie niezmienionych pozostałych elementów. Ponadto macierz $A^{(1)} = L_1AL_1^T$ jest symetryczna i dodatnio określona. Rzeczywiście,

$$\left(A^{(1)}\right)^T = (L_1AL_1^T)^T = (L_1^T)^TA^TL_1^T = L_1AL_1^T,$$

oraz dla $\vec{x} \neq 0$

$$\vec{x}^TA^{(1)}\vec{x} = \vec{x}^TL_1AL_1^T\vec{x} = (L_1^T\vec{x})^TA(L_1^T\vec{x}) > 0,$$

bo $\vec{x} \neq 0$ implikuje $L_1^T\vec{x} \neq 0$. Stąd $a_{2,2}^{(1)} = \vec{e}_2^TA^{(1)}\vec{e}_2 > 0$. Postępując tak dalej otrzymujemy

$$L_{n-1}L_{n-2}\cdots L_2L_1AL_1^TL_2^T\cdots L_{n-2}^TL_{n-1}^T = D,$$

przy czym macierz D jest diagonalna i dodatnio określona, a więc wyrazy na diagonali są dodatnie. Oznaczając

$$L = L_1^{-1}L_2^{-1}\cdots L_{n-1}^{-1}$$

dostajemy żądany rozkład.

Zauważmy, że przy praktycznej realizacji rozkładu $A = LDL^T$ wystarczy modyfikować jedynie wyrazy pod i na głównej przekątnej macierzy wyjściowej, ponieważ, jak zauważyliśmy, wszystkie kolejne macierze $A^{(k)}$ są symetryczne. Pozwala to zmniejszyć koszt kombinatoryczny o połowę do $n^3/3$ operacji arytmetycznych. Opisany sposób rozkładu macierzy $A = A^T > 0$ nosi nazwę metody Banachiewicza-Choleskiego. (Zob. Ćw. 3.9.)

U. 3.8 Rozkład trójkątno-trójkątny macierzy może być zastosowany również do policzenia wyznacznika macierzy A. Jeśli bowiem $PA = LR$ to

$$\det A = (-1)^s r_{1,1}r_{2,2}\cdots r_{n,n},$$

gdzie s jest liczbą przestawień wierszy w eliminacji.

Ćwiczenia

Ćw. 3.1 Pokazać, że jeśli algorytm rozwiązywania układu równań z macierzą pełną (lub trójkątną) nie wykorzystuje wszystkich współczynników macierzy i wektora, to istnieją macierz A i wektor $\vec{b}$ takie, że algorytm dla tych danych nie daje dokładnego rozwiązania w arytmetyce idealnej.

Ćw. 3.2 Znaleźć macierz odwrotną do macierzy trójkątnej.

Ćw. 3.3 Pokazać, że iloczyn elementarnych macierzy permutacji jest macierzą, która w każdym wierszu i w każdej kolumnie ma dokładnie jedną jedynkę. I odwrotnie, jeśli pewna macierz ma w każdym wierszu i kolumnie dokładnie jedną jedynkę to jest ona iloczynem pewnej liczby elementarnych macierzy permutacji.

Ćw. 3.4 Pokazać, że jeśli dla danej macierzy A i permutacji P istnieje co najwyżej jeden rozkład $PA = LR$ na macierz trójkątną dolną L z jedynkami na przekątnej i na macierz trójkątną górną R.

Ćw. 3.5 Pokazać, że w algorytmie eliminacji Gaussa (bez przestawień) elementy $l_{i,j}$ i $r_{i,j}$ obliczane są według wzorów

$$l_{i,j} = (a_{i,j} - \sum_{k=1}^{j-1} l_{i,k} r_{k,j})/r_{j,j}, \qquad j = 1, 2 \ldots, i-1,$$

$$r_{i,j} = a_{i,j} - \sum_{k=1}^{i-1} l_{i,k} r_{k,j}, \qquad j = i, i+1, \ldots, n,$$

dla $i = 1, 2, \ldots, n$. Wywnioskować stąd rozkład $A = LR$ z Twierdzenia 3.1.

Ćw. 3.6 Pokazać, że przy spełnieniu warunku (3.3) algorytm "przeganiania" z U. 3.6 jest wykonalny bez przestawień wierszy. Opracować algorytm rozwiązywania układu z macierzą trójdiagonalną nie spełniającą warunku (3.3).

Ćw. 3.7 Pokazać, że jeśli macierz pełna A ma dominującą przekątną, tzn.

$$2|a_{i,i}| > \sum_{j=1}^{n} |a_{i,j}|, \qquad 1 \le i \le n,$$

to jest to macierz dodatnio określona, $\vec{x}^T A \vec{x} > 0$ dla $\vec{x} \neq 0$. Ponadto, eliminacja Gaussa jest dla takich macierzy wykonalna bez przestawień wierszy.

Ćw. 3.8 Opracować algorytm eliminacyjny dla rozwiązywania układów liniowych z macierzą *Hessembega*, czyli macierzą dla której $a_{i,j} = 0$, o ile $i \ge j + 2$.

Ćw. 3.9 Opracować program rozkładający macierz symetryczną i dodatnio określoną A na iloczyn $A = LDL^T$ według metody Banachiewicza-Choleskiego z U. 3.7. Obliczenia przeprowadzać w tej samej macierzy A pozostawiając elementy na i nad diagonalą nie zmienione i otrzymując w wyniku macierz L pod główną diagonalą macierzy A, a macierz D jako dodatkowy wektor.

æ

Rozdział 4

Analiza błędów w eliminacji Gaussa

Zastanowimy się teraz, jak przebiega w arytmetyce fl_ν realizacja algorytmu eliminacji Gaussa z wyborem elementu głównego w kolumnie.

4.1 Układy z macierzą trójkątną

Najpierw pokażemy, że algorytmy rozwiązywania układów trójkątnych zaprezentowane w Rozdziale 3.1 są numerycznie poprawne. Rozpatrzmy najpierw układ trójkątny górny, $R\vec{x} = \vec{c}$. Wykorzystując fakt, że liczenie iloczynu skalarnego jest numerycznie poprawne, mamy

$$fl_\nu(x_n^*) \;=\; \frac{c_n(1+\beta_n)}{r_{n,n}(1+\alpha_{n,n})}(1+\delta_n) \;=\; \frac{c_n^\nu}{r_{n,n}^\nu}$$

i dla $i = n-1, n-2, \ldots, 1$,

$$\begin{aligned} & fl_\nu(x_i^*) \\ &\quad = \frac{\Big(c_i(1+\beta_i) \;-\; \sum_{j=i+1}^n r_{i.j}(1+\alpha_{i,j}) fl_\nu(x_j^*)(1+\gamma_{i,j})\Big)(1+\omega_i)}{r_{i,i}(1+\alpha_{i,i})(1+\delta_i)^{-1}} \\ &\quad = \frac{c_i^\nu \;-\; \sum_{j=i+1}^n r_{i,j}^\nu fl_\nu(x_j^*)}{r_{i,i}^\nu}, \end{aligned}$$

gdzie $c_i^\nu = c_i(1+\beta_i)$, $r_{i,i}^\nu = r_{i,i}(1+\alpha_{i,i})(1+\delta_i)^{-1}$, $r_{i,j}^\nu = r_{i,j}(1+\alpha_{i,j})(1+\gamma_{i,i})$, czyli $|c_i^\nu - c_i| \le \nu|c_i|$ i $|r_{i,j}^\nu - r_{i,j}| \tilde{\le} (n+1)\nu|r_{i,j}|$. Stąd otrzymane w

fl_ν rozwiązanie jest dokładnym rozwiązaniem dla danych $R^\nu = (r^\nu_{i,j})_{i,j}$ i $\vec{c}^\nu = (c^\nu_j)_j$, przy czym

$$\|R^\nu - R\|_\infty \tilde{\leq} (n+1)\,\nu\,\|R\|_\infty \qquad \text{i} \qquad \|\vec{c}^\nu - \vec{c}\|_\infty \tilde{\leq} \nu\,\|\vec{c}\|_\infty.$$

Zauważmy, że macierz zaburzona R^ν jest nieosobliwa. Powyższy algorytm jest więc numerycznie poprawny w klasie nieosobliwych macierzy trójkątnych górnych.

Zauważmy również, że jeśli wektor prawej strony $\vec{c}$ jest reprezentowany dokładnie ($\beta_i = 0$), to błąd pochodzący z odejmowania można przenieść na zaburzenia $r_{i,i}$ tak, że $\vec{c}^\nu = \vec{c}$. Fakt ten wykorzystamy później.

Przeprowadzając podobną analizę dla układu $L\vec{x} = \vec{c}$ z macierzą trójkątną dolną z jedynkami na przekątnej dostajemy, że otrzymane w fl_ν rozwiązanie jest dokładnym rozwiązaniem układu $L^\nu\vec{x} = \vec{c}^\nu$, gdzie

$$\|L^\nu - L\|_\infty \tilde{\leq} (n+1)\,\nu\,\|L\|_\infty \qquad \text{i} \qquad \|\vec{c}^\nu - \vec{c}\|_\infty \tilde{\leq} \nu\,\|\vec{c}\|_\infty.$$

4.2 Poprawność rozkładu trójkątnego

Zajmiemy się teraz jakością numeryczną rozkładu macierzy na iloczyn trójkątno-trójkątny. W poniższym twierdzeniu norma macierzy jest dowolna, ale ustalona.

Twierdzenie 4.1 *Niech $A \in \mathbf{R}^{n\times n}$ będzie macierzą nieosobliwą. Dla dostatecznie silnej arytmetyki (tzn. dla dostatecznie małego ν) rozkład trójkątno-trójkątny macierzy za pomocą eliminacji Gaussa z wyborem elementu głównego w kolumnie jest wykonalny w arytmetyce fl_ν. Otrzymane w wyniku macierze permutacji P^ν, trójkątna górna R^ν i trójkątna dolna L^ν (z jedynkami na przekątnej i elementami co do modułu nie większymi od jedności) spełniają równość*

$$P^\nu \cdot (A + E) \;=\; L^\nu \cdot R^\nu,$$

gdzie

$$\|E\| \;\leq\; K(n)\,\nu\,\|A\|,$$

a $K(n)$ jest pewną stałą niezależną od ν i A.

Dowód Załóżmy najpierw dla uproszczenia, że eliminacja jest w fl_ν wykonalna i że nie musimy dokonywać przestawień wierszy, tzn. w k-tym kroku elementem głównym jest obliczona wartość $a_{k,k}^{(k-1)}$.

Zobaczmy jak zmieniają się elementy $a_{i,j}^{(k)}$ przy realizacji algorytmu Gaussa w arytmetyce fl_ν. Dla uniknięcia nadmiaru indeksów, będziemy oznaczać przez $\tilde{w}$ wartość w obliczoną w fl_ν. Rozpatrzymy dwa przypadki, $i \le j$ i $i > j$.

Niech najpierw $i \le j$. Niech $\varepsilon_{i,j}^{(s)}$ będzie błędem bezwzględnym wytworzonym przy obliczaniu wartości $a_{i,j}^{(s)}$ mając obliczone $a_{i,j}^{(s-1)}$, $l_{i,s}$ i $a_{s,j}^{(s-1)}$. (Błąd ten oszacujemy później). Ponieważ $a_{i,j}^{(k)} = a_{i,j}^{(k-1)} - l_{i,k} a_{k,j}^{(k-1)}$, otrzymujemy

$$\begin{aligned} \tilde{a}_{i,j}^{(k)} &= \tilde{a}_{i,j}^{(k-1)} - \tilde{l}_{i,k}\tilde{a}_{k,j}^{(k-1)} + \varepsilon_{i,j}^{(k)} \qquad (4.1)\\ &= \tilde{a}_{i,j}^{(k-2)} - \tilde{l}_{i,k-1}\tilde{a}_{k-1,j}^{(k-2)} + \varepsilon_{i,j}^{(k-1)} - \tilde{l}_{i,k}\tilde{a}_{k,j}^{(k-1)} + \varepsilon_{i,j}^{(k)} \\ &= \dots \\ &= \tilde{a}_{i,j}^{(0)} + \sum_{s=1}^{k-1}\left(-\tilde{l}_{i,s}\tilde{a}_{s,j}^{(s-1)} + \varepsilon_{i,j}^{(s)}\right). \end{aligned}$$

Wobec tego, że $a_{i,j}^{(0)} = a_{i,j}$, mamy $\tilde{a}_{i,j}^{(0)} = a_{i,j} + \varepsilon_{i,j}^{(0)}$ z $|\varepsilon_{i,j}^{(0)}| \le \nu |a_{i,j}|$. Ponadto $\tilde{a}_{i,j}^{(i-1)} = \tilde{r}_{i,j}$, bo i-ty wiersz jest modyfikowany tylko w krokach od 1-go do $(i-1)$-szego. Stąd, podstawiając $k = i-1$, otrzymujemy

$$\tilde{r}_{i,j} = \tilde{a}_{i,j}^{(i-1)} = a_{i,j} + \sum_{s=0}^{i-1}\varepsilon_{i,j}^{(s)} - \sum_{s=1}^{i-1}\tilde{l}_{i,s}\tilde{r}_{s,j},$$

albo

$$a_{i,j} + e_{i,j} = \tilde{r}_{i,j} + \sum_{s=1}^{i-1}\tilde{l}_{i,s}\tilde{r}_{s,j}, \qquad (4.2)$$

gdzie $e_{i,j} = \sum_{s=0}^{i-1}\varepsilon_{i,j}^{(s)}$.

Dla $i > j$ mamy podobnie

$$\tilde{a}_{i,j}^{(j-1)} = a_{i,j} + \varepsilon_{i,j}^{(0)} + \sum_{s=1}^{j-1}\left(-\tilde{l}_{i,s}\tilde{a}_{s,j}^{(s-1)} + \varepsilon_{i,j}^{(s)}\right). \qquad (4.3)$$

Wobec tego, że

$$\tilde{l}_{i,j} = \left(\frac{\tilde{a}_{i,j}^{(j-1)}}{\tilde{a}_{j,j}^{(j-1)}}\right)(1 + \delta_{i,j}),$$

otrzymujemy

$$\tilde{a}_{i,j}^{(j-1)} \;=\; \tilde{l}_{i,j}\tilde{a}_{j,j}^{j-1} + \varepsilon_{i,j}^{(j)}, \qquad |\varepsilon_{i,j}^{(j)}| \;\leq\; \nu|\tilde{a}_{i,j}^{(j-1)}|,$$

a stąd, (4.3) i z faktu, że $\tilde{a}_{s,j}^{(s-1)} = \tilde{r}_{i,j}$,

$$a_{i,j} \,+\, e_{i,j} \;=\; \sum_{s=1}^{j} \tilde{l}_{i,s}\tilde{r}_{s,j}, \tag{4.4}$$

gdzie $e_{i,j} = \sum_{s=0}^{j} \varepsilon_{i,j}^{(s)}$.

Z (4.2) i (4.4) wynika równość

$$(A \,+\, E) \;=\; \tilde{L} \cdot \tilde{R} \tag{4.5}$$

z macierzą $E = (e_{i,j})$. Ponadto elementy $\tilde{l}_{i,j}$ są co do modułu nie większe od jedności, ponieważ $|\tilde{a}_{i,j}^{(j-1)}| \leq |\tilde{a}_{j,j}^{(j-1)}|$ (zob. Ćw. 1.2).

Aby zakończyć dowód, trzeba jeszcze oszacować błąd $|e_{i,j}|$. W tym celu zauważmy, że dla $k \geq 1$

$$\tilde{a}_{i,j}^{(k)} \;=\; \Big(\tilde{a}_{i,j}^{(k-1)} - \tilde{l}_{i,k}\tilde{a}_{k,j}^{(k-1)}(1+\omega_1)\Big)(1+\omega_2),$$

$\omega_1, \omega_2 \leq \nu$. Stąd i z (4.1) otrzymujemy

$$\begin{aligned} \varepsilon_{i,j}^{(k)} \;&=\; \tilde{a}_{i,j}^{(k)} \,-\, \tilde{a}_{i,j}^{(k-1)} \,+\, \tilde{l}_{i,k}\tilde{a}_{k,j}^{(k-1)} \;=\; \tilde{a}_{i,j}^{(k)} \,-\, \frac{\tilde{a}_{i,j}^{(k)}}{1+\omega_2} \,+\, \omega_1\tilde{l}_{i,k}\tilde{a}_{k,j}^{(k-1)} \\ &=\; \frac{\omega_2}{1+\omega_2}\tilde{a}_{i,j}^{(k)} \,+\, \omega_1\tilde{l}_{i,k}\tilde{a}_{k,j}^{(k-1)} \end{aligned}$$

oraz

$$|\varepsilon_{i,j}^{(k)}| \;\leq\; \frac{\nu}{1-\nu}(|\tilde{a}_{i,j}^{(k)}| + |\tilde{a}_{k,j}^{(k-1)}|).$$

Niech teraz $G_0 = \max_{i,j}|a_{i,j}|$,

$$G_k \;=\; \max_{i,j}|\tilde{a}_{i,j}^{(k)}|,$$

oraz $G = \max_{0\leq k\leq n-1} G_k$. Dla $i \leq j$ mamy

$$\begin{aligned} |e_{i,j}| \;&\leq\; \sum_{s=0}^{i-1}|\varepsilon_{i,j}^{(s)}| \;\leq\; \frac{\nu}{1-\nu}\Big(G_0 + \sum_{s=1}^{i-1}(G_{s-1}+G_s)\Big) \\ &=\; \frac{\nu}{1-\nu}\Big(G_{i-1} + 2\sum_{s=0}^{i-2}G_s\Big) \;\leq\; \frac{\nu}{1-\nu}(2i-1)G, \end{aligned}$$

a dla $i > j$

$$\begin{aligned} |e_{i,j}| &\leq \sum_{s=0}^{j} |\varepsilon_{i,j}^{(s)}| \leq \frac{\nu}{1-\nu}\Big(G_0 + \sum_{s=1}^{j}(G_{s-1}+G_s)\Big) \\ &= \frac{\nu}{1-\nu}\Big(G_j + 2\sum_{s=0}^{j-1} G_s\Big) \leq \frac{\nu}{1-\nu}(2j+1)G. \end{aligned}$$

W obu przypadkach

$$|e_{i,j}| \leq \frac{2\nu}{1-\nu} nG. \tag{4.6}$$

Łatwo zauważyć, że $\tilde{a}_{i,j}^{(1)} \tilde{\leq} (|a_{i,j}| + |a_{k,j}|)(1+2\nu)$ oraz

$$|\tilde{a}_{i,j}^{(k)}| \leq (|\tilde{a}_{i,j}^{(k-1)}| + |\tilde{a}_{k,j}^{(k-1)}|)(1+\nu),$$

co implikuje $G_k \tilde{\leq} 2G_{k-1}$, a w konsekwencji

$$G \tilde{\leq} 2^{n-1} \max_{i,j} |a_{i,j}|.$$

Ostatnia nierówność oraz (4.6) dają

$$|e_{i,j}| \tilde{\leq} n 2^n \nu \max_{i,j} |a_{i,j}|,$$

a ponieważ dla dowolnej macierzy $B \in \mathbf{R}^{n\times n}$ mamy

$$\frac{1}{n}\|B\|_\infty \leq \max_{i,j} |a_{i,j}| \leq \|B\|_\infty,$$

to

$$\|E\|_\infty \leq K(n)\, \nu\, \|A\|_\infty,$$

gdzie $K(n)$ jest na poziomie $n^2 2^n$. Z równoważności norm w $\mathbf{R}^{n\times n}$ wynika, że podobna nierówność zachodzi dla dowolnej, innej niż $\|\cdot\|_\infty$ normy, ale być może z inną stałą $K(n)$.

Oczywiście, ewentualne permutacje wierszy nie zmieniają tych oszacowań, a powodują jedynie przemnożenie lewej strony równości (4.5) przez pewną macierz permutacji.

Aby zakończyć dowód, trzeba jeszcze pokazać, że dla dostatecznie małych ν rozkład jest wykonalny, co jest równoważne temu, że macierz $(A+E)$ jest nieosobliwa. Rzeczywiście, weźmy $\nu > 0$ spełniające

$$\nu \, K(n) \, \|A\|_\infty \|A^{-1}\|_\infty \; < \; 1. \tag{4.7}$$

Jeśliby macierz $(A+E)$ była osobliwa to dla pewnego niezerowego wektora $\vec{x}$ mielibyśmy $(A+E)\vec{x} = A(I+A^{-1}E)\vec{x} = 0$, a w konsekwencji $\|A^{-1}E\vec{x}\|_\infty = \|\vec{x}\|_\infty$ i

$$1 \; \le \; \|A^{-1}E\|_\infty \; \le \; \|A^{-1}\|_\infty \|E\|_\infty \; \le \; K(n)\nu \|A\|_\infty \|A^{-1}\|_\infty,$$

co przeczy (4.7). □

Zauważmy, że warunek (4.7) na wykonalność rozkładu w fl_ν zależy od macierzy A poprzez wielkość

$$\mathrm{cond}(A) \; = \; \|A\| \, \|A^{-1}\|.$$

Wielkość tą nazywa się *uwarunkowaniem macierzy.* Z Twierdzenia 4.1 oraz warunku (4.7) wynika ważny wniosek o numerycznej poprawności rozkładu w klasie macierzy o uwarunkowaniu wspólnie ograniczonym przez pewną stałą.

Wniosek 4.1 *Niech $M > 0$. W klasie macierzy nieosobliwych A takich, że*

$$cond(A) \; = \; \|A\| \, \|A^{-1}\| \; \le \; M$$

rozkład $PA = LR$ jest numerycznie poprawny. Dokładniej, dla $\nu < 1/(MK(n))$ rozkład w fl_ν jest wykonalny, a otrzymane macierze P^ν, L^ν i R^ν pochodzą z dokładnego rozkładu macierzy $(A+E)$, gdzie $\|E\| \le K(n)\nu\|A\|$. □

4.3 Poprawność rozwiązywania układu

Przypomnijmy, że algorytm eliminacji Gaussa rozwiązywania układów równań $A\vec{x} = \vec{b}$ polega na rozkładzie macierzy na iloczyn $PA = LR$, a następnie na rozwiązaniu dwóch układów trójkątnych (albo, równoważnie, na sprowadzeniu układu do postaci trójkątnej i rozwiązaniu go). Numeryczna poprawność każdego z tych etapów algorytmu daje nam również numeryczną poprawność całego algorytmu.

Twierdzenie 4.2 *Niech $M > 0$. Algorytm eliminacji Gaussa z wyborem elementu głównego w kolumnie zastosowany do rozwiązania układu równań*

$$A\vec{x} = \vec{b}$$

jest numerycznie poprawny w klasie danych wektorów $\vec{b} \in \mathbf{R}^n$ oraz macierzy nieosobliwych $A \in \mathbf{R}^{n\times n}$ spełniających

$$cond(A) \le M.$$

Dowód Dla dostatecznie silnej arytmetyki, rozkład macierzy A daje w fl_ν $P_1(A + E_1) = L_1 R_1$, gdzie $\|E_1\| \le K_1\nu\|A\|$. W drugim kroku, otrzymane w fl_ν rozwiązanie $\vec{y}_1$ układu $L_1\vec{y} = P_1\vec{b}$ jest dokładnym rozwiązaniem układu $(L_1 + E_2)\vec{y} = P_1(\vec{b} + \vec{e})$, gdzie $\|E_2\| \le K_2\nu\|L_1\|$ oraz $\|\vec{e}\| \le K_3\nu\|\vec{b}\|$. Z kolei w ostatnim kroku algorytmu, rozwiązując układ $R_1\vec{x} = \vec{y}_1$ dostajemy w fl_ν rozwiązanie $\vec{x}^\nu$, które jest dokładnym rozwiązaniem układu $(R_1 + E_3)\vec{x} = \vec{y}_1$, gdzie $\|E_3\| \le K_3\nu\|R_1\|$. (Zauważmy, że wektor $\vec{y}_1$ nie jest zaburzony, bo jest on reprezentowany dokładnie, zob. Rozdział 4.1). Stąd otrzymane w fl_ν rozwiązanie $\vec{x}^\nu$ jest dokładnym rozwiązaniem układu

$$A^\nu\vec{x} = \vec{b}^\nu,$$

gdzie $\vec{b}^\nu = \vec{b} + \vec{e}$, a $A^\nu = A + E$ spełnia równanie

$$P_1(A + E) = (L_1 + E_2)\,(R_1 + E_3).$$

Aby zakończyć dowód, należy teraz oszacować $\|E\|$. Ponieważ $P_1(A + E_1) = L_1R_1$, mamy

$$P_1E = P_1E_1 + L_1E_3 + E_2R_2 + E_2E_3.$$

Stąd

$$\begin{aligned}
\|E\|_\infty &\le \|E_1\|_\infty + \|L_1\|_\infty\|E_3\|_\infty + \|E_2\|_\infty\|R_2\|_\infty + \|E_2\|_\infty\|E_3\|_\infty \\
&\mathrel{\tilde{\le}} \nu K_1\|A\|_\infty + \|L_1\|_\infty\nu K_3\|R_1\|_\infty + \|R_1\|_\infty\nu K_2\|L_1\|_\infty \\
&= \nu\Big(K_1\|A\|_\infty + (K_2 + K_3)\|L_1\|_\infty\|R_1\|_\infty\Big).
\end{aligned}$$

Ponieważ elementy macierzy L_1 są co do modułu nie większe od jedności, mamy $\|L_1\|_\infty \le n$. Jak zauważyliśmy wcześniej, mamy też $\|R_1\|_\infty \le$

$n2^{n-1}\|A\|_\infty$. Przyjmując $K = K_1 + (K_2 + K_3)n^2 2^{n-1}$ ostatecznie otrzymujemy

$$\|E\|_\infty \le K\,\nu\,\|A\|_\infty,$$

czyli numeryczną poprawność, ponieważ stała K nie zależy od ν i A. □

4.4 Uwarunkowanie macierzy, a błąd w fl_ν

Pokazaliśmy, że eliminacja Gaussa jest numerycznie poprawna w klasie macierzy $A \in \mathbf{R}^{n\times n}$ takich, że

$$\mathrm{cond}(A) \le M,$$

gdzie $M < \infty$ jest jakąkolwiek stałą niezależną od A. Okazuje się, że wielkość uwarunkowania macierzy, cond(A), ma też zasadniczy wpływ na błąd zadania rozwiązywania układu równań. Rzeczywiście, mamy bowiem następujące twierdzenie. (Poniżej norma wektorowa jest dowolna, ale ustalona, a norma macierzowa jest przez nią indukowana, zob. U. 1.2.)

Twierdzenie 4.3 *Niech E i $\vec{e}$ będą zaburzeniami odpowiednio macierzy A i wektora $\vec{b}$ takimi, że*

$$\|E\| \le K_1\nu\|A\| \qquad i \qquad \|\vec{e}\| \le K_2\nu\|\vec{b}\|,$$

Jeśli

$$K_1\nu\, cond(A) < 1$$

to układ zaburzony $(A+E)\vec{x} = (\vec{b}+\vec{e})$ ma jednoznaczne rozwiązanie $\vec{z}^*$ spełniające

$$\|\vec{z}^* - \vec{x}^*\| \le (K_1 + K_2)\,\nu\,\frac{cond(A)}{1 - K_1\nu\, cond(A)}\|\vec{x}^*\|.$$

Dowód Zauważmy najpierw, że jeśli F jest macierzą taką, że $\|F\| < 1$ to macierz $(I - F)$ (gdzie I jest macierzą identycznościową) jest nieosobliwa oraz

$$\|(I-F)^{-1}\| \;\le\; \frac{1}{1-\|F\|}. \qquad (4.8)$$

Rzeczywiście, gdyby $(I-F)$ była osobliwa to istniałby niezerowy wektor $\vec{x}$ taki, że $(I-F)\vec{x} = 0$, co implikuje $\|F\vec{x}\|/\|\vec{x}\| = 1$ i w konsekwencji $\|F\| \ge 1$. Aby pokazać (4.8) zauważmy, że

$$\begin{aligned} 1 &= \|I\| = \|(I-F)(I-F)^{-1}\| \\ &\ge \|(I-F)^{-1}\| - \|F\|\,\|(I-F)^{-1}\| \\ &= (1-\|F\|)\,\|(I-F)^{-1}\|, \end{aligned}$$

skąd bezpośrednio wynika (4.8).

Po podstawieniu $F = -A^{-1}E$ mamy teraz

$$\|F\| \;\le\; \|A^{-1}\|\,\|E\| \;\le\; K_1\nu\|A\|\,\|A^{-1}\| \;<\; 1,$$

co wobec równości $A + E = A(I + A^{-1}E)$ daje, że macierz $(A + E)$ jest nieosobliwa i układ zaburzony ma jednoznaczne rozwiązanie $\vec{z}^*$. Przedstawmy to rozwiązanie w postaci $\vec{z}^* = \vec{x}^* + (\vec{z}^* - \vec{x}^*)$. Rozpisując układ zaburzony i wykorzystując równość $A\vec{x}^* = \vec{b}$ otrzymujemy, że $(A+E)(\vec{z}^* - \vec{x}^*) = \vec{e} - E\vec{x}^*$, czyli

$$\vec{z}^* - \vec{x}^* \;=\; (I + A^{-1}E)^{-1}A^{-1}(\vec{e} - E\vec{x}^*),$$

a stąd

$$\begin{aligned} \|\vec{z}^* - \vec{x}^*\| &\le \|(I + A^{-1}E)^{-1}\|\,\|A^{-1}\|\,(\|\vec{e}\| + \|E\|\,\|\vec{x}^*\|) \\ &\le \frac{\|A^{-1}\|}{1 - K_1\nu\|A\|\,\|A^{-1}\|}\Big(K_2\nu\|\vec{b}\| + K_1\nu\|A\|\,\|\vec{x}^*\|\Big) \\ &\le \frac{\|A\|\,\|A^{-1}\|}{1 - K_1\nu\|A\|\,\|A^{-1}\|}(K_1 + K_2)\,\nu\,\|\vec{x}^*\|, \end{aligned}$$

co kończy dowód. $\Box$

Podsumowując Twierdzenia 4.2 i 4.3 otrzymujemy wniosek, który jest końcowym wynikiem tego rozdziału.

Wniosek 4.2 *Niech A będzie macierzą nieosobliwą. Dla dostatecznie silnej arytmetyki,*

$$\nu\,\overline{K}(n)\,cond(A)\ \ll\ 1$$

(gdzie $\overline{K}(n)$ jest pewną stłą niezależną od A i ν), eliminacja Gaussa z wyborem elementu głównego w kolumnie, zastosowana do rozwiązania układu $A\vec{x}=\vec{b}$, jest w fl_ν wykonalna i daje wynik $fl_\nu(\vec{x}^*)$ spełniający nierówność

$$\|fl_\nu(\vec{x}^*)\ -\ \vec{x}^*\|\ \tilde{\leq}\ \overline{K}(n)\,\nu\,cond(A)\,\|\vec{x}^*\|. \qquad \square$$

Uwagi i uzupełnienia

U. 4.1 Jak zauważyliśmy w dowodzie Twierdzenia 4.1 stała kumulacji $K(n)$ zależy zasadniczo od wzrostu maksymalnego elementu G_k w macierzach $A^{(k)}$ powstających w kolejnych krokach eliminacji. Okazuje się, że uzyskane, pesymistyczne oszacowanie $G=\max_k G_k\leq 2^{n-1}G_0$ jest praktycznie nie spotykane, choć teoretycznie może być osiągnięte. Przykładem jest macierz

$$\begin{pmatrix} 1 & 0 & 0 & \cdots & 0 & 1\\ -1 & 1 & 0 & & \vdots & 1\\ -1 & -1 & 1 & & \vdots & 1\\ & & & \ddots & 0 & \vdots\\ & & & & 1 & 1\\ -1 & -1 & -1 & \cdots & -1 & 1 \end{pmatrix},$$

dla której $G_k=2^kG_0$, $0\leq k\leq n-1$, oraz $G=2^{n-1}G_0$.

U. 4.2 Eliminacja Gaussa *bez* przestawień wierszy jest numerycznie poprawna, gdy element maksymalny w kolejnych macierzach $A^{(k)}$ wzrasta w sposób niezależny od A. Na przykład, gdy $G_k\leq mG_{k-1}$ to $G\leq m^{k-1}G_0$ i w Twierdzeniu 4.1 mamy $K(n)=2n^2m^{k-1}$. Ma to miejsce wtedy gdy A jest symetryczna i dodatnio określona, albo ma dominującą główną przekątną, zob. U. 4.3 oraz Ćw. 4.5 i 4.6.

U. 4.3 Dla macierzy symetrycznych i dodatnio określonych, $A=A^T>0$, eliminacja Gaussa jest wykonalna bez przestawień wierszy, zob. U. 3.7. W klasie tych macierzy eliminacja bez przestawień wierszy jest też numerycznie poprawna, a więc w szczególności algorytm Banachiewicza-Choleskiego

jest numerycznie popeawny. Wykażemy to pokazując, że maksymalny element w kolejnych macierzach $A^{(k)}$ wzrasta co najwyżej dwukrotnie, tzn. $G_k \le 2G_{k-1}$, $1 \le k \le n$. W tym celu wykorzystamy fakt, że dla dowolnej symetrycznej i dodatnio określonwj macierzy $B = (b_{i,j})$ mamy $b_{i,i} > 0$, oraz spełniona jest nierówność

$$b_{i,j}^2 < b_{i,i} b_{j,j}, \qquad \forall i, j \tag{4.9}$$

(zob. Ćw. 4.1). Jak wykazaliśmy w U. 3.7, każda z macierzy $A^{(k)}$ jest symetryczna i dodatnio określona, Stąd

$$\begin{aligned} |a_{i,j}^{(k)}| &= |a_{i,j}^{(k-1)} - l_{i,k} a_{k,j}^{(k-1)}| \\ &\le |a_{i,j}^{(k-1)}| + \frac{|a_{i,k}^{(k-1)}|}{|a_{k,k}^{(k-1)}|} |a_{k,j}^{(k-1)}| \\ &\le \sqrt{a_{i,i}^{(k-1)} a_{j,j}^{(k-1)}} + \frac{\sqrt{a_{i,i}^{(k-1)} a_{k,k}^{(k-1)}} \sqrt{a_{k,k}^{(k-1)} a_{j,j}^{(k-1)}}}{a_{k,k}^{(k-1)}} \\ &= 2\sqrt{a_{i,i}^{(k-1)} a_{j,j}^{(k-1)}} \le 2 \max(a_{i,i}^{(k-1)}, a_{j,j}^{(k-1)}), \end{aligned}$$

a w konsekwencji $G_k \le 2G_{k-1}$.

Ćwiczenia

Ćw. 4.1 Wykazać, że macierz symetryczna 2×2

$$\begin{pmatrix} a & c \\ c & b \end{pmatrix}$$

jest dodatnio określona wtedy i tylko wtedy, gdy $a > 0$ i $ab > c^2$. Wywnioskować stąd nierównosc (4.9) dla macierzy symetrycznych i dodatnio określonych dowolnego wymiaru. Ponadto, największy co do modułu element takiej macierzy leży na głównej diagonali.

Ćw. 4.2 Wykazać, że macierz A jest dodatnio określona wtedy i tylko wtedy gdy dla każdego $\vec{x} \in \mathbf{R}^n$ wektory $A\vec{x}$ i $\vec{x}$ tworzą w $\mathbf{R}^n$ kąt ostry.

Ćw. 4.3 Pokazać, że jeśli eliminację Gaussa z wyborem elementu głównego w kolumnie zastosujemy do macierzy trójdiagonalnej, to wzrost elementu maksymalnego macierzy nie będzie zależał od n. Dokładniej,

$$\max_{i,j,k} |a_{i,j}^{(k)}| \;\leq\; 2 \max_{i,j} |a_{i,j}|.$$

Ćw. 4.4 Pokazać, że dla macierzy Hessenberga ($a_{i,j} = 0$ dla $i \geq j+2$) eliminacja Gaussa z wyborem elementu głównego w kolumnie daje

$$\max_{i,j,k} |a_{i,j}^{(k)}| \;\leq\; (k+1) \max_{i,j} |a_{i,j}|.$$

Ćw. 4.5 Pokazać numeryczną poprawność algorytmu przeganiania z U. 3.6.

Ćw. 4.6 Wykazać numeryczną poprawność eliminacji Gaussa bez przestawień wierszy dla macierzy z dominującą przekątną, tzn. gdy

$$2|a_{i,i}| \;>\; \sum_{j=1}^{n} |a_{i,j}|, \qquad 1 \leq i \leq n.$$

Wskazówka. Zauważyć, że $\max_{i,j} |a_{i,j}^{(k)}| \;\leq\; 2 \max_{i,j} |a_{i,j}^{(k-1)}|$.

Ćw. 4.7 Jeśli

$$(A+E)\vec{z} \;=\; \vec{b}, \tag{4.10}$$

gdzie $\|E\|_p \leq K\nu\|A\|_p$, to oczywiście dla residuum $\vec{r} = \vec{b} - A\vec{z}$ mamy

$$\|\vec{r}\|_p \;\leq\; K\nu\|A\|_p\|\vec{z}\|_p. \tag{4.11}$$

Pokazać, że dla $p = 1, 2, \infty$ zachodzi też twierdzenie odwrotne, tzn. jeśli spełniony jest warunek (4.11) to istnieje macierz pozornych zaburzeń E taka, że $\|E\|_p \leq K\nu\|A\|_p$ oraz spełniona jest równość (4.10).
Wskazówka. Rozpatrzyć $E = \vec{r}(sgn\, z_i)^T_{1\leq i\leq n}/\|\vec{z}\|_1$ dla $p = 1$, $E = \vec{r}\vec{z}^T/\|\vec{z}\|_2^2$ dla $p = 2$, oraz $E = \vec{r}(sgn\, z_k)\vec{e}_k^T/\|\vec{z}\|_\infty$ dla $p = \infty$, gdzie k jest indeksem dla którego $|z_k| = \|\vec{z}\|_\infty$.

Rozdział 5

Zadanie wygładzania liniowego

W tym rozdziale zajmiemy się zadaniem wygładzania liniowego, nazywanym też liniowym zadaniem najmniejszych kwadratów. Jest ono uogólnieniem zadania rozwiązywania kwadratowych układów równań liniowych do przypadku, gdy układ jest nadokreślony.

5.1 Układ normalny

Niech A będzie daną macierzą o m wierszach i n kolumnach, $A \in \mathbf{R}^{m \times n}$, taką, że

$$m \geq n = \mathrm{rank}(A),$$

albo równoważnie, taką że jej wektory kolumny są liniowo niezależne. Niech także dany będzie wektor $\vec{b} \in \mathbf{R}^m$. Jasne jest, że wtedy układ równań $A\vec{x} = \vec{b}$ nie zawsze ma rozwiązanie - mówimy, że układ jest *nadokreślony.*

Zadanie wygładzania liniowego polega na znalezieniu wektora $\vec{x}^* \in \mathbf{R}^n$, który minimalizuje *wektor residualny* $\vec{r} = \vec{b} - A\vec{x}$ w normie drugiej, tzn.

$$\|\vec{b} - A\vec{x}^*\|_2 = \min_{\vec{x} \in \mathbf{R}^n} \|\vec{b} - A\vec{x}\|_2.$$

Przykład 5.1 Przypuśćmy, że dla pewnej funkcji $f : [a, b] \to \mathbf{R}$ obserwujemy jej wartości f_i (dokładne lub zaburzone) w punktach t_i,

$1 \le i \le m$. Funkcję tą chcielibyśmy przybliżyć inną funkcją w należącą do pewnej n wymiarowej przestrzeni liniowej W, np. przestrzeni wielomianów stopnia mniejszego niż n. Jakość przybliżenia mierzymy wielkością

$$\sum_{i=1}^{m} (f_i - w(t_i))^2. \tag{5.1}$$

Wybierając pewną bazę $(w_j)_{j=1}^n$ w W i rozwijając w w tej bazie, $w(t) = \sum_{j=1}^n c_j w_j(t)$, sprowadzamy problem do minimalizacji

$$\sum_{i=1}^{m} \left(f_i - \sum_{j=1}^{n} c_j w_j(t_i) \right)^2$$

względem c_j, a więc do zadania wygładzania liniowego. Rzeczywiście, kładąc $A = (a_{i,j}) \in \mathbf{R}^{m\times n}$ z $a_{i,j} = w_j(t_i)$, $\vec{b} = (f_i)_{i=1}^m$ i $\vec{x} = (c_j)_{j=1}^n$, wielkość (5.1) jest równa $\|\vec{b} - A\vec{x}\|_2^2$.

Lemat 5.1 *Zadanie wygładzania liniowego ma jednoznaczne rozwiązanie $\vec{x}^*$, które spełnia układ równań*

$$A^T A\vec{x} = A^T \vec{b}. \tag{5.2}$$

Dowód Niech $P \subset \mathbf{R}^m$ będzie obrazem A jako odwzorowania liniowego z $\mathbf{R}^n$ w $\mathbf{R}^m$,

$$P = \{ A\vec{x} : \vec{x} \in \mathbf{R}^n \}.$$

Ponieważ kolumny macierzy A są liniowo niezależne, tworzą one bazę w P. Stąd $\dim(P) = n$ i odwzorowanie $A : \mathbf{R}^n \to P$ jest różnowartościowe. Ponadto przestrzeń $\mathbf{R}^m$ z normą drugą jest przestrzenią unitarną. Residuum jest więc minimalizowane dla wektora $\vec{x}^* \in \mathbf{R}^n$ takiego, że $A\vec{x}^*$ jest rzutem prostopadłym wektora $\vec{b}$ na podprzestrzeń P. (Przypomnijmy, że skończony wymiar P zapewnia, że taki rzut istnieje i jest wyznaczony jednoznacznie.) Równoważnie można powiedzieć, że residuum $\vec{b} - A\vec{x}^*$ jest prostopadłe do P,

$$(A\vec{x})^T(\vec{b} - A\vec{x}^*) = 0, \qquad \forall \vec{x} \in \mathbf{R}^n,$$

albo

$$x^T(\vec{A}^T\vec{b} - A^T A\vec{x}^*) = 0. \qquad \forall \vec{x} \in \mathbf{R}^n.$$

Otrzymaliśmy więc, że wektor $A^T\vec{b} - A^T A\vec{x}^*$ jest prostopadły w $\mathbf{R}^n$ do każdego innego wektora. Ponieważ jedynym wektorem o tej własności jest wektor zerowy, to $A^T A\vec{x}^* = A^T\vec{b}$. □

Zauważmy, że jeśli macierz A jest kwadratowa, $m = n$, to rozwiązaniem jest $\vec{x}^* = A^{-1}\vec{b}$ i residuum jest zerem. Zadanie wygładzania liniowego jest więc uogólnieniem rozwiązywania kwadratowych układów równań liniowych.

Równanie (5.2) nazywa się układem *normalnym*. Może ono nam sugerować sposób rozwiązania zadania wygładzania liniowego. Wystarczy bowiem pomnożyć macierz A^T przez A i rozwiązać układ normalny. Zauważmy ponadto, że macierz $A^T A$ jest symetryczna i dodatnio określona, bo $(A^T A)^T = A^T A$ i dla $\vec{x} \neq 0$ mamy $\vec{x}^T(A^T A)\vec{x} = (A\vec{x})^T(A\vec{x}) = \|A\vec{x}\|_2 > 0$, przy czym ostatnia nierówność wynika z faktu, że kolumny macierzy A są liniowo niezależne i dlatego $A\vec{x} \neq \vec{0}$. Przy mnożeniu A^T przez A wystarczy więc obliczyć tylko elementy na głównej przekątnej i pod nią, a do rozwiązania równania z macierzą $A^T A$ można zastosować algorytm Banachiewicza-Choleskiego opisany w U. 3.7. Jak łatwo się przekonać, koszt takiego algorytmu wynosi $n^2(k + n/3)$, przy czym dominuje koszt mnożenia obliczenia macierzy $A^T A$.

Ma on jednak pewne wady. Mnożenie macierzy powoduje w fl_ν powstanie "po drodze" dodatkowych błędów, które mogą nawet zmienić rząd macierzy. Na przykład, dla macierzy

$$A = \begin{pmatrix} 1 & 1 & 1 & 1 \\ \varepsilon & & & \\ & \varepsilon & & \\ & & \varepsilon & \\ & & & \varepsilon \end{pmatrix}$$

mamy

$$A^T A = \begin{pmatrix} 1+\varepsilon^2 & 1 & 1 & 1 \\ 1 & 1+\varepsilon^2 & 1 & 1 \\ 1 & 1 & 1+\varepsilon^2 & 1 \\ 1 & 1 & 1 & 1+\varepsilon^2 \end{pmatrix}.$$

Jeśli $\varepsilon^2 < \nu$ to $fl_\nu(1+\varepsilon^2) = 1$, co implikuje $\operatorname{rank}(fl_\nu(A^T A)) = 1$, podczs gdy $\operatorname{rank}(fl_\nu(A)) = 4$.

Poniżej przedstawimy inną metodę rozwiązywania zadania wygładzania liniowego, która oparta jest na specjalnych przekształceniach zwanych odbiciami Hauseholdera.

5.2 Odbicia Hauseholdera

Dla danego wektora $\vec{w} \in \mathbf{R}^m$ o normie $\|\vec{w}\|_2 = \sqrt{\vec{w}^T\vec{w}} = 1$, *odbicie (macierz) Hauseholdera* zdefiniowane jest jako

$$H = I - 2\vec{w}\vec{w}^T.$$

Zauważmy, że

$$H\vec{x} = \vec{x} - 2(\vec{w}^T\vec{x})\vec{w},$$

a ponieważ $(\vec{w}^T\vec{x})\vec{w} = \langle\vec{x},\vec{w}\rangle_2\vec{w}$ jest rzutem prostopadłym $\vec{x}$ na kierunek wektora $\vec{w}$ ($\langle\cdot,\cdot\rangle_2$ oznacza iloczyn skalarny), to $H\vec{x}$ jest odbiciem lustrzanym wektora $\vec{x}$ względem hiperpłaszczyzny (wymiaru $m-1$) prostopadłej do $\vec{w}$.

Odbicia Hauseholdera są przekształceniami nieosobliwymi spełniającymi

$$H^{-1} = H = H^T.$$

Rzeczywiście, ponieważ $\vec{w}$ ma normę jednostkową, mamy

$$H^2 = (I - 2\vec{w}\vec{w}^T)^2 = I - 4\vec{w}\vec{w}^T + 4\vec{w}(\vec{w}^T\vec{w})\vec{w}^T = I,$$

oraz

$$H^T = (I - 2\vec{w}\vec{w}^T)^T = I - 2(\vec{w}^T)^T\vec{w}^T = I.$$

W szczególności H jest więc przekształceniem *ortogonalnym*, $H^{-1} = H^T$, czyli nie zmienia długości wektora,

$$\|H\vec{x}\|_2 = \sqrt{(H\vec{x})^T(H\vec{x})} = \sqrt{\vec{x}^T(H^TH)\vec{x}} = \sqrt{\vec{x}^T\vec{x}} = \|\vec{x}\|_2.$$

Odbicia Hauseholdera zastosujemy do przeprowadzenia danego wektora $\vec{a} \neq \vec{0}$ na kierunek innego niezerowego wektora, powiedzmy $\vec{e}$, tzn.

$$H\vec{a} = (I - 2\vec{w}\vec{w}^T)\vec{a} = \alpha\,\vec{e}.$$

Załóżmy dla uproszczenia, że $\|\vec{e}\|_2 = 1$. Aby wyznaczyć H zauważmy, że

$$\vec{w} \;=\; \frac{\vec{a} - \alpha\vec{e}}{2(\vec{w}^T\vec{a})},$$

a ponieważ $\alpha = \pm\|\vec{a}\|_2$ i $\|\vec{w}\|_2 = 1$ to

$$\vec{w} \;=\; \frac{\vec{a} \mp \|\vec{a}\|_2\vec{e}}{\|\vec{a} \mp \|\vec{a}\|_2\vec{e}\|_2}.$$

W szczególności, jeśli $\vec{e} = \vec{e}_1$ jest pierwszym wersorem to powyższe wzory dają

$$H \;=\; I \;-\; \frac{\vec{u}\vec{u}^T}{\gamma},$$

gdzie

$$u_i \;=\; \begin{cases} a_1 \mp \|\vec{a}\|_2 & i = 1, \\ a_i & 2 \le i \le m, \end{cases}$$

oraz

$$\begin{aligned} \gamma \;&=\; \frac{1}{2}\|\vec{u}\|_2^2 \;=\; \frac{1}{2}\Big((a_1 \mp \|\vec{a}\|_2)^2 + \sum_{i=2}^m a_i^2\Big) \\ &=\; \frac{1}{2}\Big(\sum_{i=1}^m a_i^2 \;+\; \|\vec{a}\|_2^2 \;\mp\; 2a_1\|a\|_2\Big) \;=\; \|a\|_2^2 \;\mp\; a_1\|a\|_2. \end{aligned}$$

Otrzymaliśmy dwa odbicia Hauseholdera przekształcające dany wektor $\vec{a}$ na kierunek pierwszego wersora, w zależności od wybranego znaku przy $\|\vec{a}\|_2$. Ustalimy ten znak na plus gdy $a_1 \ge 0$, oraz na minus gdy $a_1 < 0$, co pozwoli na obliczenie u_1 i γ z małym błędem względem w fl_ν. Wtedy bowiem mamy

$$u_1 \;=\; \begin{cases} a_1 + \|a\|_2 & a_1 \ge 0, \\ a_1 - \|a\|_2 & a_1 < 0, \end{cases}$$

oraz $\gamma = \|\vec{a}\|_2^2 + |a_1|\,\|\vec{a}\|_2$, czyli zawsze dodajemy liczby tych samych znaków. Ponadto pierwsza współrzędna wektora $H\vec{a}$ jest równa $-\|\vec{a}\|_2$ dla $a_1 \ge 0$, oraz $+\|\vec{a}\|_2$ dla $a_1 < 0$.

5.3 Algorytm dla zadania wygładzania

Odbić Hauseholdera można użyć do rozkładu macierzy $A \in \mathbf{R}^{m\times n}$ na iloczyn ortogonalno-trójkątny.

Niech $A = (\vec{a}_1, \vec{a}_2, \ldots, \vec{a}_n)$, gdzie $\vec{a}_j$ są wektorami-kolumnami macierzy A. Wybierzmy pierwsze odbicie Hauseholdera $H_1 = I_m - \vec{u}_1\vec{u}_1^T/\gamma_1$ tak, aby przekształcało pierwszy wektor-kolumnę macierzy A na kierunek $\vec{e}_1$. Efektem pomnożenia macierzy A z lewej strony przez H_1 będzie wtedy macierz

$$A^{(1)} \;=\; (\vec{a}_1^{(1)}, \ldots, \vec{a}_n^{(1)}) \;=\; (H_1\vec{a}_1, \ldots, H_1\vec{a}_n),$$

w której pierwsza kolumna $\vec{a}_1^{(1)}$ ma niezerową tylko pierwszą współrzędną. W następnym kroku wybieramy drugie przekształcenie Hauseholdera $\bar{H}_2 = I_{m-1} - \vec{v}_2\vec{v}_2^T/\gamma_2$ wymiaru $m-1$ tak, aby przeprowadzało wektor $(a_{i,2}^{(1)})_{i=2}^m$ na kierunek pierwszego wersora w $\mathbf{R}^{m-1}$. Rozszerzając $\vec{v}_2 \in \mathbf{R}^{m-1}$ do wektora $\vec{u}_2 \in \mathbf{R}^m$ przez dodanie zera jako pierwszej współrzędnej, $\vec{u}_2 = (0, \vec{v}_2)^T$, otrzymujemy przekształcenie (macierz) Hauseholdera $H_2 = I_m - \vec{u}_2\vec{u}_2^T/\gamma_2$ w $\mathbf{R}^m$ postaci

$$H_2 \;=\; \begin{pmatrix} 1 & \vec{0}^T \\ \vec{0} & \bar{H}_2 \end{pmatrix}.$$

Pomnożenie macierzy $A^{(1)}$ z lewej strony przez H_2 spowoduje teraz wyzerowanie drugiej kolumny macierzy pod elementem $a_{2,2}^{(1)}$, przy czym pierwszy wiersz i pierwsza kolumna pozostaną niezmienione. Postępując tak dalej n razy (albo $n-1$ razy gdy $m = n$) otrzymujemy

$$H_n H_{n-1} \cdots H_2 H_1 A \;=\; R,$$

gdzie $R \in \mathbf{R}^{m\times n}$ jest uogólnioną macierzą trójkątną górną, tzn. $r_{i,j} = 0$ dla $i > j$. Stąd, podstawiając $Q = H_1 H_2 \cdots H_n$, dostajemy rozkład macierzy na iloczyn ortogonalno-trójkątny

$$A \;=\; Q \cdot R. \tag{5.3}$$

Rzeczywiście, macierz $Q \in \mathbf{R}^{m\times m}$ jest ortogonalna, bo

$$\begin{aligned} Q^{-1} &= (H_1 H_2 \cdots H_n)^{-1} \;=\; H_n^{-1} \cdots H_2^{-1} H_1^{-1} \\ &= H_n^T \cdots H_2^T H_1^T \;=\; (H_1 H_2 \cdots H_n)^T \;=\; Q^T. \end{aligned}$$

Dyspunując rozkładem (5.3) zadanie wygładzania liniowego można rozwiązać następująco. Ponieważ mnożenie przez macierz ortogonalną nie zmienia normy drugiej wektora, mamy

$$\begin{aligned}\|\vec{r}\|_2 &= \|\vec{b} - A\vec{x}\|_2 \;=\; \|\vec{b} - QR\vec{x}\|_2 \\ &= \|Q(Q^T\vec{b} - R\vec{x})\|_2 \;=\; \|\vec{c} - R\vec{x}\|_2,\end{aligned}$$

gdzie $\vec{c} = Q^T\vec{b} = H_n \cdots H_2 H_1 \vec{b}$. Rozbijając wektor $\vec{c}$ na $\vec{c} = (\vec{c}_I, \vec{c}_{II})^T$, gdzie $\vec{c}_I \in \mathbf{R}^n$ i $\vec{c}_{II} \in \mathbf{R}^{m-n}$, oraz macierz R na

$$R = \begin{pmatrix} R_I \\ 0 \end{pmatrix},$$

gdzie $R_I \in \mathbf{R}^{n \times n}$ jest macierzą trójkątną górną, a 0 jest macierzą zerową wymiaru $(m-n) \times n$, otrzymujemy

$$\|\vec{r}\|_2^2 \;=\; \|\vec{c}_I - R_I\vec{x}\|_2^2 \;+\; \|\vec{c}_{II}\|_2^2.$$

Rozwiązanie $\vec{x}^*$ zadania wygładzania jest więc rozwiązaniem układu liniowego trójkątnego,

$$\vec{x}^* \;=\; R_I^{-1}\vec{c}_I,$$

oraz $\|\vec{r}^*\|_2 = \|\vec{b} - A\vec{x}^*\|_2 = \|\vec{c}_{II}\|_2$.

Zastanówmy się nad praktyczną realizacją tego algorytmu. Każde z kolejnych przekształceń Hauseholdera H_k wyznaczamy przez obliczenie γ_k oraz współrzędnych wektora $\vec{u}_k$. Wektor ten ma tylko $m-k+1$ współrzędnych niezerowych, a ponadto $u_{k,i} = a_{i,k}^{(k-1)}$ dla $k+1 \leq i \leq m$. Dzięki takiej reprezentacji H_k, mnożenia $H_k\vec{x}$ możemy dla dowolnego $\vec{x}$ realizować według wzoru

$$(H_k\vec{x})_i \;=\; x_i \;-\; s\, u_{k,i},$$

gdzie $s = \vec{u}_k^T\vec{x}/\gamma_k$.

Uwzględnizjąc obecność zerowych elementów w $\vec{u}_k$, przejście od macierzy $A^{(k-1)}$ do $A^{(k)}$ kosztuje rzędu $4(m-k+1)(n-k)$ operacji arytmetycznych i obliczenie jednego pierwiastka kwadratowego. Cały rozkład $A = QR$ kosztuje więc rzędu (dla dużych m i n)

$$\sum_{k=1}^{n} 4(m-k+1)(n-k) \;\approx\; \frac{4}{3}n^3 + 2n^2(m-n) \;=\; 2n^2(m-n/3)$$

operacji arytmetycznych i n pierwiastków kwadratowych. Zauważmy, że w przypadku $m = n$, a więc dla kwadratowego układu równań, koszt ten wynosi $(4/3)n^3$ i jest dwa razy większy od kosztu eliminacji Gaussa.

Uwagi i uzupełnienia

U. 5.1 Pokazaliśmy, że rozwiązaniem zadania wygładzania liniowego jest wektor $\vec{x}^* = A^+\vec{b}$, gdzie

$$A^+ = (A^T A)^{-1} A^T.$$

Macierz $A^+ \in \mathbf{R}^{n\times m}$ nazywa się macierzą *pseudoodwrotną* do $A \in \mathbf{R}^{m\times n}$, $\mathrm{rank}(A) = n$. Dla nieosobliwych macierzy kwadratowych mamy oczywiście $A^+ = A^{-1}$, ponieważ wtedy $\vec{x}^* = A^{-1}$.

U. 5.2 Pokażemy, że każdą macierz $A^{m\times n}$, $\mathrm{rank}(A) = n$, można rozłożyć na iloczyn

$$A = U\,\Sigma\,V^T, \tag{5.4}$$

gdzie $U \in \mathbf{R}^{m\times m}$ i $V \in \mathbf{R}^{n\times n}$ są macierzami ortogonalnymi, a $\Sigma \in \mathbf{R}^{m\times n}$ jest macierzą diagonalną (tzn. $\sigma_{i,j} = 0$ dla $i \neq j$) o dodatnich wyrazach $\sigma_{i,i}$.

Ponieważ macierz $A^T A$ jest symetryczna i dodatnio określona, znane twierdzenie z algebry liniowej mówi, że istnieje w $\mathbf{R}^n$ baza ortonormalna $(\vec{\xi}_j)_{j=1}^n$ wektorów własnych tej macierzy, a odpowiadające im wartości własne są rzeczywiste i dodatnie, tzn.

$$\langle \vec{\xi}_i, \vec{\xi}_j \rangle_2 = \begin{cases} 0 & i \neq j, \\ 1 & i = j, \end{cases}$$

oraz $(A^T A)\vec{\xi}_j = \lambda_j \vec{\xi}_j$, gdzie

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n > 0.$$

Zauważmy, że wektory $\vec{\eta}_i = \lambda_i^{-1/2}(A\vec{\xi}_i)$ są ortonormalne w $\mathbf{R}^m$, bowiem

$$\begin{aligned} \langle \vec{\eta}_i, \vec{\eta}_j \rangle_2 &= (\lambda_i \lambda_j)^{-1/2} \langle A\vec{\xi}_i, A\vec{\xi}_j \rangle_2 \\ &= (\lambda_i \lambda_j)^{-1/2} \langle (A^T A)\vec{\xi}_i, \vec{\xi}_j \rangle_2 \\ &= (\lambda_i \lambda_j)^{-1/2} \lambda_i \langle \vec{\xi}_i, \vec{\xi}_j \rangle_2 = \begin{cases} 0 & i \neq j, \\ 1 & i = j. \end{cases} \end{aligned}$$

Wektory $\vec{\eta}_i$, $1 \le i \le n$, można uzupełnić $m - n$ dodatkowymi wektorami tak, aby cały układ $(\vec{\eta}_i)_{i=1}^m$ był bazą ortonormalną w $\mathbf{R}^m$. Zdefiniujmy teraz macierze ortogonalne U o kolumnach $\vec{\eta}_i$, $1 \le i \le m$, oraz V o kolumnach $\vec{\xi}_j$, $1 \le j \le n$. Bezpośredni rachunek pokazuje, że macierz $U^T AV$ jest diagonalna z wyrazami na przekątnej $\sqrt{\lambda_j}$. Stąd $A = U\Sigma V^T$ z $\sigma_{j,j} = \sqrt{\lambda_j}$, $1 \le j \le n$.

Liczby $\sqrt{\lambda_j}$ nazywa się *wartościami szczególnymi* macierzy A, a rozkład (5.4) rozkładem macierzy według wartości szczególnych.

Zauważmy jeszcze, że jeśli $A = U\Sigma V^T$ to $A^+ = V\Sigma^+ U^T$ oraz $\Sigma^+ \in \mathbf{R}^{n\times m}$ jest macierzą diagonalną z wyrazami na diagonali równymi $1/\sqrt{\lambda_j}$. Rozwiązanie zadania wygładzania liniowego można więc zapisać jako

$$\vec{x}^* = V\,\Sigma^+ U^T \vec{b}.$$

U. 5.3 Przedstawimy teraz jedną z możliwych implementacji algorytmu Hauseholdera rozkładu macierzy na iloczyn $A = QR$. Po wykonaniu poniższego programu wyrazy $r_{i,j}$ macierzy R dla $1 \le i \le j$ zostaną zapamiętane na miejscach $a[i,j]$, współrzędne $u_{j,i}$ wektora $\vec{u}_j$ dla $j+1 \le i \le m$ na miejscach $a[i,k]$, a współrzędna $u_{j,j}$ i liczba γ_j odpowiednio na $u[j]$ i $gam[j]$, $1 \le j \le n$.

```
for k := 1 to n do
begin
    { obliczanie kolejnego odbicia Hauseholdera }
    norm2 := 0.0;
    for l := k to m do
    begin
        aa := a[l,k];
        norm2 := norm2 + aa * aa
    end;
    norm := sqrt(norm2);
    aa := a[k,k];
    if (aa ≥ 0.0) then
    begin
        uu := aa + norm;
        akk := −norm
    end else
    begin
        uu := aa − norm;
        akk := norm
```

```
        end;
        gamma := norm2 + abs(aa) * norm;
        u[k] := uu;
        gam[k] := gamma;
        { modyfikacja kolumn macierzy }
        a[k,k] := akk;
        for  j := k+1  to  n  do
        begin
            s := uu * a[k,j];
            for  l := k+1  to  m  do
                s := s + a[l.k] * a[l,j];
            s := s/gamma;
            a[k,j] := a[k,j] - s * uu;
            for  l := k+1  to  m  do
                a[l,j] := a[l,j] - s * a[l,k]
        end;
    end.
```

U. 5.4 Można pokazać, że przedstawiony algorytm Hauseholdera rozkładu macierzy na iloczyn ortogonalno-trójkątny jest numerycznie poprawny. To znaczy, otrzymane w fl_ν macierz trójkątna górna R^ν i ortogonalna Q^ν (reprezentowana przez n wektorów $\vec{u}_k$ i liczb γ_k) spełniają $(A+E) = Q^\nu R^\nu$, gdzie $\|E\| \leq K(m,n)\nu\|A\|$.

U. 5.5 Zadanie wygładzania liniowego można również rozwiązać stosując innego rodzaju rozkład ortogonalno-trójkątny macierzy $A \in \mathbf{R}^{m\times n}$; mianowicie przez zastosowanie *ortogonalizacji Grama-Schmidta* do wektorów kolumn $\vec{a}_j$ macierzy A. W wyniku otrzymujemy ciąg wektorów $\vec{q}_j$, $1 \leq j \leq n$, tworzący układ ortonormalny w $\mathbf{R}^m$, oraz spełniający

$$\mathrm{span}\{\vec{q}_1, \vec{q}_2, \ldots, \vec{q}_j\} \;=\; \mathrm{span}\{\vec{a}_1, \vec{a}_2, \ldots, \vec{a}_j\}, \qquad 1 \leq j \leq n. \tag{5.5}$$

Zauważmy, że jeśli z wektorów $\vec{q}_j$ stworzymy macierz $Q = (\vec{q}_1, \vec{q}_2, \ldots, \vec{q}_n) \in \mathbf{R}^{m\times n}$, to (5.5) implikują istnienie kwadratowej macierzy trójkątnej górnej $R \in \mathbf{R}^{n\times n}$ takiej, że

$$A \;=\; Q \cdot R.$$

(W odróżnieniu od efektu działania algorytmu Hauseholdera, macierz Q nie jest tu kwadratowa, ale za to kwadratowa jest macierz R.)

Wektory ortonormalne $\vec{q}_j$ oraz współczynniki $r_{i,j}$ macierzy R można wyznaczyć z układu równań

$$\vec{a}_j = \sum_{s=1}^{j} r_{s,j}\vec{q}_s, \qquad 1 \leq j \leq n.$$

Mnożąc j-te równanie skalarnie przez $\vec{q}_i$, $1 \leq i \leq j$, otrzymujemy $r_{i,j} = \langle \vec{a}_j, \vec{q}_i \rangle_2 = \vec{q}_i^T \vec{a}_j$, a stąd wzory rekurencyjne

for $j := 1$ **to** n **do**
begin
  for $i := 1$ **to** $j-1$ **do** $r_{i,j} := \langle \vec{q}_i, \vec{a}_j \rangle_2$;
  $\vec{p}_j := \vec{a}_j - \sum_{s=1}^{j-1} r_{s,j}\vec{q}_s$;
  $r_{j,j} := \|\vec{p}_j\|_2$;
  $\vec{q}_j := \vec{p}_j / r_{j,j}$
end.

Dysponując rozkładem $A = QR$, rozwiązanie $\vec{x}^*$ zadania wygładzania wyznaczamy z równania $R\vec{x} = Q^T\vec{b}$. Rzeczywiście, układ $(\vec{q}_j)_{j=1}^n$ można formalnie uzupełnić do układu ortonormalnego w $\mathbf{R}^m$ dodając do niego pewne wektory $\vec{q}_j$ dla $n+1 \leq j \leq m$. Tworząc macierz ortogonalną $\overline{Q} = (\vec{q}_1, \ldots, \vec{q}_m)$ i rozszerzając macierz $R \in \mathbf{R}^{n\times n}$ do macierzy $\overline{R} \in \mathbf{R}^{m\times n}$ przez dodanie do niej $(m-n)$ zerowych wierszy, otrzymujemy $A = \overline{Q}\,\overline{R}$, a więc rozkład taki jak w algorytmie Hauseholdera. Postępując dalej tak, jak w analizie algorytmu Hauseholdera, otrzymujemy żądany wynik.

U. 5.6 Okazuje się, że algorytm ortogonalizacyjny Grama-Schmidta z U. 5.5 ma niedobre własności numeryczne, gdy wektory-kolumny macierzy wyjściowej A są "słabo liniowo niezależne"; tzn. otrzymane w fl_ν wektory $\vec{q}_j^{(1)}$ mogą być wtedy "słabo" ortogonalne. W takim wypadku należy stosować podwójną ortogonalizację, najpierw do wektorów $\vec{a}_j$, a potem do $\vec{q}_j^{(1)}$, $1 \leq j \leq n$. (Zob. Ćw. 5.7.)

Ćwiczenia

Ćw. 5.1 Pokazać, że dla macierzy pseudoodwrotnej A^+ do danej macierzy $A \in \mathbf{R}^{m\times n}$, $\mathrm{rank}(A) = n$, macierz $A^+A = I_n$ jest identycznością w $\mathbf{R}^{n\times n}$, natomiast

$$AA^+ = \begin{pmatrix} I_n & 0 \\ 0 & 0 \end{pmatrix} \in \mathbf{R}^{m\times m},$$

czyli AA^+ jest rzutem prostopadłym na podprzestrzeń rozpiętą na pierwszych n wersorach w $\mathbf{R}^m$.

Ćw. 5.2 Niech $A \in \mathbf{R}^{m\times n}$. Uzasadnić, że jądro macierzy A^T,

$$\ker(A^T) \;=\; \{\, \vec{y} \in \mathbf{R}^m :\; A^T\vec{y} = \vec{0}\,\},$$

jest podprzestrzenią prostopadłą do obrazu A,

$$\mathrm{im}(A) \;=\; \{\, A\vec{x} \in \mathbf{R}^m :\; \vec{x} \in \mathbf{R}^n\,\}.$$

Ćw. 5.3 Pokazać, że macierze A^TA i AA^T mają takie same niezerowe wartości własne, a podprzestrzenie własne im odpowiadające mają ten sam wymiar.

Ćw. 5.4 Uzasadnić, że dana macierz kwadratowa $Q \in \mathbf{R}^{m\times m}$ jest ortogonalna wtedy i tylko wtedy gdy jej kolumny (wiersze) tworzą bazę ortonormalną w $\mathbf{R}^m$.

Ćw. 5.5 Niech $\vec{u}$ będzie niezerowym wektorem w $\mathbf{R}^m$ oraz $\gamma = \|\vec{u}\|_2^2/2$. Uzasadnić, że algorytm obliczania

$$H\vec{x} \;=\; \vec{x} \;-\; s\vec{u}, \qquad s \;=\; (\vec{u}^T\vec{x})/\gamma,$$

według powyższego wzoru, jest numerycznie poprawny ze względu na dany wektor $\vec{x} \in \mathbf{R}^m$.

Ćw. 5.6 Policzyć złożoność algorytmu ortogonalizacyjnego opisanego w U. 5.5 rozwiązania zadania wygładzania liniowego.

Ćw. 5.7 Zastosujmy ortogonalizację Grama-Schmidta do dwóch wektorów liniowo niezależnych $\vec{a}$ i $\vec{b}$ o normach $\|\vec{a}\|_2 = 1 = \|\vec{b}\|_2$. Załóżmy dla uproszczenia, że w obliczeniach *jedynie* iloczyn skalarny tych wektorów $s = \langle\vec{a},\vec{b}\rangle_2$ liczy się z błędem, $fl_\nu(s) \;=\; s(1+\varepsilon)$, gdzie $|\varepsilon|$ jest dodatnie i na poziomie ν. Pokazać, że wtedy dla otrzymanej w wyniku ortogonalizacji unormowanej pary wektorów $\vec{a}$ i $\vec{c}$ mamy

$$\langle\vec{a},\vec{c}\rangle_2 \;=\; \frac{-\varepsilon s}{\sqrt{1 - s^2(1-\varepsilon^2)}}.$$

Wywnioskować stąd, że gdy $\vec{a}$ i $\vec{b}$ są “prawie liniowo zależne”, to $\vec{a}$ i $\vec{c}$ są dalekie od ortogonalnych.

Ćw. 5.8 Zaprogramować algorytm Grama-Schmidta z U. 5.5 (z podwójną ortogonalizacją) rozwiązujący zadanie wygładzania.

Ćw. 5.9 Zaprogramować algorytm obliczający macierz odwrotną do danej nieosobliwej macierzy $A \in \mathbf{R}^{n \times n}$ wykorzystujący rozkład Hauseholdera $A = QR$.

æ

Rozdział 6

Interpolacja wielomianowa

Dotychczas rozpatrywaliśmy zadania, w których danymi są ciągi liczb rzeczywistych. Teraz zajmiemy się zadaniami, w których danymi są funkcje o wartościach rzeczywistych. Pierwszym z nich jest zadanie interpolacji wielomianowej.

6.1 Sformułowanie zadania interpolacji

Niech $D \subset \mathbf{R}$ i niech F będzie pewnym zbiorem funkcji $f : D \to \mathbf{R}$. Niech $x_0, x_1, \ldots, x_n$ będzie ustalonym zbiorem parami różnych punktów z D, zwanych później *węzłami*.

Powiemy, że wielomian w *interpoluje* funkcję $f \in F$ w węzłach x_j, gdy

$$w(x_j) = f(x_j), \qquad 0 \le j \le n.$$

Oznaczmy przez Π_n przestrzeń liniową wielomianów stopnia co najwyżej n o współczynnikach rzeczywistych,

$$\Pi_n = \{ w(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0 : \ a_j \in \mathbf{R}, 0 \le j \le n \}.$$

Lemat 6.1 *Dla dowolnej funkcji $f : D \to \mathbf{R}$ istnieje dokładnie jeden wielomian $w_f \in \Pi_n$ interpolujący f w węzłach x_j, $0 \le j \le n$.*

Dowód Wybierzmy w Π_n dowolną bazę wielomianów φ_j, $0 \le j \le n$,

$$\Pi_n = \mathrm{span}\{ \varphi_0, \varphi_1, \ldots, \varphi_n \}.$$

Wtedy każdy wielomian z Π_n można jednoznacznie przedstawić w postaci rozwinięcia względem wybranej bazy. Warunkiem koniecznym i dostatecznym na to, aby wielomian $w_f(\cdot) = \sum_{j=0}^n c_j \varphi_j(\cdot)$ interpolował f jest spełnienie układu $n+1$ równań liniowych

$$\sum_{j=0}^n c_j \varphi_j(x_i) \;=\; f(x_i), \qquad 0 \le i \le n,$$

z $n+1$ niewiadomymi c_j, który w postaci macierzowej wygląda następująco:

$$\begin{pmatrix} \varphi_0(x_0) & \varphi_1(x_0) & \cdots & \varphi_n(x_0) \\ \varphi_0(x_1) & \varphi_1(x_1) & \cdots & \varphi_n(x_1) \\ \vdots & & & \\ \varphi_0(x_n) & \varphi_1(x_n) & \cdots & \varphi_n(x_n) \end{pmatrix} \begin{pmatrix} c_0 \\ c_1 \\ \vdots \\ c_n \end{pmatrix} = \begin{pmatrix} f(x_0) \\ f(x_1) \\ \vdots \\ f(x_n) \end{pmatrix}. \tag{6.1}$$

Aby wykazać, że układ ten ma jednoznaczne rozwiązanie wystarczy, aby wektor zerowy był jedynym rozwiązaniem układu jednorodnego. Rzeczywiście, układ jednorodny odpowiada interpolacji danych zerowych, $f(x_i) = 0$, $\forall i$. Istnienie niezerowego rozwiązania byłoby więc równoważne istnieniu niezerowego wielomianu stopnia nie większego od n, który miałby $n+1$ różnych zer x_i, co jest niemożliwe. $\square$

Zadanie znalezienia dla danej funkcji f jej wielomianu interpolacyjnego stopnia co najwyżej n jest więc dobrze zdefiniowane, tzn. rozwiązanie istnieje i jest wyznaczone jednoznacznie. Zauważmy, że wielomian interpolacyjny w_f jako taki nie może być wynikiem obliczeń w naszym modelu obliczeniowym, możemy natomiast wyznaczyć jego współczynniki c_j w wybranej bazie.

Definicja 6.1 *Niech $(\varphi_j)_{j=0}^n$ będzie bazą w przestrzeni Π_n wielomianów stopnia co najwyżej n. Zadanie (obliczeniowe) interpolacji wielominowej polega na obliczeniu dla danej funkcji f współczynników c_j takich, że wielomian*

$$w_f(\cdot) \;=\; \sum_{j=0}^n c_j \varphi_j(\cdot) \tag{6.2}$$

interpoluje f w punktach x_j, $0 \le j \le n$.

6.2 Wybór bazy wielomianowej

Można powiedzieć, że trudność zadania interpolacji jest zdeterminowana przez trudność rozwiązania układu (6.1), ta zaś zależy w istotny sposób od wybranej bazy $(\varphi_j)_{j=0}^n$. W naturalny sposób powstaje więc problem wyboru "wygodnej" bazy w Π_n. Rozpatrzymy trzy bazy: Lagrange'a, potęgową i Newtona.

Baza Lagrange'a (kanoniczna)

Zdefiniujmy dla $0 \le j \le n$ wielomiany

$$l_j(x) \;=\; \frac{(x-x_0)(x-x_1)\cdots(x-x_{j-1})(x-x_{j+1})\cdots(x-x_n)}{(x_j-x_0)(x_j-x_1)\cdots(x_j-x_{j-1})(x_j-x_{j+1})\cdots(x_j-x_n)}. \tag{6.3}$$

Zauważmy, że każdy z l_j jest stopnia dokładnie n oraz

$$l_j(x_i) \;=\; \begin{cases} 0 & i \neq j, \\ 1 & i = j. \end{cases}$$

Stąd łatwo widać, że wielomiany te stanowią bazę w Π_n, którą nazywamy bazą Lagrange'a. Macierz układu (6.1) jest identycznością i w konsekwencji $c_j = f(x_j)$, $\forall j$. Wielomian interpolacyjny dla funkcji f można więc w tym przypadku zapisać jako

$$w_f(\cdot) \;=\; \sum_{j=0}^n f(x_j) l_j(\cdot).$$

Koszt kombinatoryczny rozwiązania zadania interpolacji jest przy tym zerowy.

Przypuśćmy, że chcielibyśmy obliczyć wartość wielomianu interpolacyjnego w_f w punkcie x różnym od x_j, $0 \le j \le n$. Podstawiając

$$a_j \;=\; \frac{1}{(x_j-x_0)(x_j-x_1)\cdots(x_j-x_{j-1})(x_j-x_{j+1})\cdots(x_j-x_n)}$$

oraz $a(x) = (x-x_0)\cdots(x-x_n)$ mamy

$$w_f(x) \;=\; a(x) \sum_{j=0}^n \frac{a_j f(x_j)}{x-x_j}, \tag{6.4}$$

albo

$$w_f(x) \;=\; \frac{\sum_{j=0}^n q_j(x)f(x_j)}{\sum_{j=0}^n q_j(x)},$$

gdzie $q_j(x) = a_j/(x - x_j)$. W ostatniej równości wykorzystaliśmy fakt, że $a(x) \equiv (\sum_{j=0}^n q_j(x))^{-1}$, co wynika z (6.4) po podstawieniu $f \equiv 1$.

Baza potęgowa (naturalna)

Znacznie prościej można obliczyć wartość wielomianu interpolacyjnego, (a także jego pochodnych), gdy jest on dany w najczęściej używanej bazie potęgowej, $\varphi_j(x) = x^j$, $\forall j$. Jeśli bowiem

$$w_f(x) \;=\; a_0 + a_1 x + \cdots + a_n x^n,$$

to również

$$w_f(x) \;=\; (\cdots(a_n x + a_{n-1})x + a_{n-2})x + \cdots + a_1)x + a_0,$$

co sugeruje zastosowanie następującego *schematu Hornera* do obliczenia $w_f(x)$:

$v_n \;:=\; a_n;$
for $\;j \;:=\; n-1\;$ **downto** $\;0\;$ **do**
$\qquad v_j \;:=\; v_{j+1} * x \;+\; a_j,$

Po wykonaniu tego algorytmu $w_f(x) = v_0$. Schemat Hornera wymaga wykonania tylko n mnożeń i n dodawań. Ma on również głębszy sens, zob. Ćw. 6.2.

Zauważmy jednak, że w przypadku bazy potęgowej macierz $(x_i^j)_{i,j=0}^n$ układu (6.1) jest pełna. Jest to tzw. *macierz Vandermonde'a*. Obliczenie współczynników wielomianu interpolacyjnego w bazie potęgowej bezpośrednio z tego układu, stosując jedną ze znanych nam już metod, kosztowałoby rzędu n^3 operacji arytmetycznych.

Baza Newtona

Rozwiązaniem pośrednim, które łączy prostotę obliczenia współczynników z prostotą obliczenia wartości $w_f(x)$ jest wybór bazy Newtona,

$$\begin{aligned} p_0(x) &= 1, \\ p_j(x) &= (x-x_0)(x-x_1)\cdots(x-x_{j-1}), \qquad 1 \le j \le n. \end{aligned}$$

W tym przypadku współczynniki rozwinięcia wielomianu interpolacyjnego będziemy oznaczać przez b_j,

$$w_f \;=\; \sum_{j=0}^{n} b_j p_j.$$

Zwróćmy od razu uwagę na ważną własność bazy Newtona. Jeśli $w_{f,j} \in \Pi_j$ jest wielomianem interpolacyjnym dla funkcji f opartym na węzłach $x_0, x_1, \ldots, x_j$, $0 \le j \le n$, to $w_{f,0} = b_0$ oraz

$$w_{f,j} \;=\; w_{f,j-1} \,+\, b_j p_j, \qquad 1 \le j \le n. \tag{6.5}$$

Wartość $w_f(x)$ można obliczyć stosując prostą modyfikację algorytmu Hornera:

$v_n \;:=\; b_n;$
for $j \;:=\; n-1$ **downto** 0 **do**
  $v_j \;:=\; v_{j+1} * (x - x_j) \,+\, b_j.$

Ponadto układ równań (6.1) jest trójkątny dolny (tzn. $p_j(x_i) = 0$ dla $i < j$), co pozwala na obliczenie b_j algorytmem z Rozdziału 3.1 o koszcie n^2,

$b_0 \;:=\; f(x_0);$
for $j \;:=\; 1$ **to** n **do**
  $b_j \;:=\; \Big(f(x_j) \,-\, \sum_{s=0}^{j-1} b_s p_s(x_j)\Big)\,/p_j(x_j).$

Zwykle jednak do obliczania współczynników b_j stosuje się nieco inny algorytm, który teraz predstawimy.

6.3 Algorytm różnic dzielonych

Różnicę dzieloną funkcji f opartą na różnych węzłach $t_0, t_1, \ldots, t_s$, gdzie $s \ge 1$, definiuje się indukcyjnie jako

$$f(t_0, t_1, \ldots, t_s) \;=\; \frac{f(t_1, t_2, \ldots, t_s) \,-\, f(t_0, t_1, \ldots, t_{s-1})}{t_s \,-\, t_0}. \tag{6.6}$$

Zachodzi następujące ważne twierdzenie.

Twierdzenie 6.1 *Współczynniki b_j wielomianu interpolacyjnego Newtona dla danej funkcji f dane są przez różnice dzielone f w węzłach $x_0, x_1, \ldots, x_j$, tzn.*

$$b_j \;=\; f(x_0, x_1, \ldots, x_j), \qquad 0 \le j \le n.$$

Dowód Dla $0 \le i \le j \le n$, oznaczmy przez $w_{i,j}$ wielomian z Π_{j-i} interpolujący f w węzłach $x_i, x_{i+1}, \ldots, x_j$. Wtedy ma miejsce następująca równość ($i < j$):

$$w_{i,j}(x) \;=\; \frac{(x - x_i)w_{i+1,j}(x) \;-\; (x - x_j)w_{i,j-1}(x)}{x_j \;-\; x_i}, \qquad \forall x. \qquad (6.7)$$

Aby ją pokazać wystarczy, że prawa strona tej równości, którą oznaczymy przez $v(x)$, przyjmuje wartości $f(x_s)$ dla $x = x_s$, $i \le s \le j$. Rzeczywiście, jeśli $i + 1 \le s \le j - 1$ to

$$v(x_s) \;=\; \frac{(x_s - x_i)f(x_s) - (x_s - x_j)f(x_s)}{x_j - x_i} \;=\; f(x_s).$$

Ponadto

$$v(x_i) \;=\; \frac{-(x_i - x_j)}{x_j - x_i} f(x_i) \;=\; f(x_i),$$

oraz podobnie $v(x_j) = f(x_j)$. Stąd v jest wielominem z Π_{j-i} interpolującym f w węzłach x_s, $i \le s \le j$, czyli $w_{i,j} = v$.

Dalej postępujemy indukcyjnie ze względu na stopień n wielomianu interpolacyjnego. Dla $n = 0$ mamy oczywiście $b_0 = f(x_0)$. Niech $n \ge 1$. Ponieważ, jak łatwo zauważyć,

$$w_{0,n}(x) \;=\; w_{0,n-1}(x) + b_n p_n(x),$$

z założenia indukcyjnego mamy $b_j = f(x_0, \ldots, x_j)$ dla $0 \le j \le n - 1$. Aby pokazać podobną równość dla b_n, wykorzystamy równość (6.7) z $i = 0$ i $j = n$, która ma wtedy postać

$$w_{0,n}(x) \;=\; \frac{(x - x_0)w_{1,n}(x) - (x - x_n)w_{0,n-1}(x)}{x_n - x_0}.$$

Zauważmy teraz, że b_n jest współczynnikiem przy x^n w wielomianie $w_{0,n}$. Z założenia indukcyjnego wynika, że współczynniki przy x^{n-1} w

wielomianach $w_{1,n}$ i $w_{0,n-1}$ są ilorazami różnicowymi opartymi odpowiednio na węzłach $x_1, \ldots, x_n$ i $x_0, \ldots, x_{n-1}$. Stąd

$$b_n = \frac{f(x_1, \ldots, x_n) - f(x_0, \ldots, x_{n-1})}{x_n - x_0} = f(x_0, x_1, \ldots, x_n),$$

co kończy dowód. □

Różnicę dzieloną $f(x_0, x_1, \ldots, x_n)$ można łatwo obliczyć na podstawie wartości $f(x_j)$, $0 \le j \le n$, budując następującą tabelkę:

$$\begin{array}{llllll}
x_0 & f(x_0) & & & & \\
x_1 & f(x_1) & f(x_0, x_1) & & & \\
x_2 & f(x_2) & f(x_1, x_2) & f(x_0, x_1, x_2) & & \\
\vdots & \vdots & \vdots & \vdots & \ddots & \\
x_n & f(x_n) & f(x_{n-1}, x_n) & f(x_{n-2}, x_{n-1}, x_n) & \cdots & f(x_0, x_1, \ldots, x_n).
\end{array}$$

Zauważmy przy tym, że "po drodze" obliczamy $f(x_i, x_{i+1}, \ldots, x_j)$ dla wszystkich $0 \le i < j \le n$, a więc w szczególności również interesujące nas różnice dzielone $f(x_0, x_1, \ldots, x_j)$. Stąd i z Twierdzenia 6.1 natychmiast wynika algorytm obliczania współczynników b_j wielomianu interpolacyjnego Newtona. Po wykonaniu następującego programu,

```
for  j := 0  to  n  do  b[j] := f(x[j]);
for  j := 1  to  n  do
for  k := n  downto  j  do
      b[k] := (b[k] - b[k-1])/(x[k] - x[k-j]),
```

współczynniki b_j zostaną umieszczone w tablicy $b[j]$.

6.4 Przypadek węzłów wielokrotnych

Uogólnieniem rozpatrzonego zadania interpolacji jest zadanie interpolacji *Hermite'a*. Zakładamy, że oprócz (różnych) węzłów x_j dane są również ich krotności n_j, $0 \le j \le k$, przy czym $\sum_{j=0}^{k} n_j = n+1$. Należy skonstruować wielomian $w_f \in \Pi_n$ taki, że

$$w_f^{(i)}(x_j) = f^{(i)}(x_j) \qquad \text{dla} \quad 0 \le i \le n_j - 1, 0 \le j \le k.$$

Oczywiście zakładamy przy tym, że odpowiednie pochodne funkcji f istnieją.

Lemat 6.2 *Zadanie interpolacji Hermite'a ma jednoznaczne rozwiązanie.*

Dowód Istnienie i jednoznaczność rozwiązania można uzasadnić tak samo jak w przypadku węzłów jednokrotnych. Przedstawiając wielomian w dowolnej bazie otrzymujemy układ $n+1$ równań z $n+1$ niewiadomymi, który dla zerowej prawej strony ma jedynie rozwiązanie zerowe. Inaczej bowiem istniałby wielomian niezerowy stopnia nie większego niż n, który miałby zera o łącznej krotności większej niż n. $\square$

Nas oczywiście interesuje konstrukcja wielomianu w_f. W tym celu ustawimy węzły x_j w ciąg

$$(\bar{x}_0, \bar{x}_1, \ldots, \bar{x}_n) \;=\; (\underbrace{x_0, \ldots, x_0}_{n_0}, \underbrace{x_1, \ldots, x_1}_{n_1}, \ldots, \underbrace{x_k, \ldots, x_k}_{n_k})$$

i zdefiniujemy uogólnioną bazę Newtona w Π_n jako

$$\begin{aligned} p_0(x) &= 1, \\ p_j(x) &= (x-\bar{x}_0)(x-\bar{x}_1)\cdots(x-\bar{x}_{j-1}), \qquad 1 \le j \le n. \end{aligned}$$

Uogólnimy również pojęcie różnicy dzielonej na węzły powtarzające się kładąc

$$f(\bar{x}_i, \bar{x}_{i+1}, \ldots, \bar{x}_j) \;=\; \frac{f^{(j-i)}(\bar{x}_i)}{(j-i)!}$$

dla $\bar{x}_i = \bar{x}_{i+1} = \cdots = \bar{x}_j$, oraz

$$f(\bar{x}_i, \bar{x}_{i+1}, \ldots, \bar{x}_j) \;=\; \frac{f(\bar{x}_{i+1}, \ldots, \bar{x}_j) - f(\bar{x}_i, \ldots, x_{j-1})}{\bar{x}_j - \bar{x}_i}$$

dla $\bar{x}_i \ne \bar{x}_j$. Zauważmy, że przy tej definicji różnice $f(\bar{x}_i, \ldots, \bar{x}_j)$ możemy łatwo obliczyć stosując schemat podobny do tego z przypadku węzłów jednokrotnych.

Twierdzenie 6.2 *Współczynniki b_j wielomianu interpolacyjnego Hermite'a w bazie Newtona,*

$$w_f(\cdot) \;=\; \sum_{j=0}^{n} b_j p_j(\cdot),$$

dane są przez odpowiednie różnice dzielone, tzn.

$$b_j \;=\; f(\bar{x}_0, \bar{x}_1, \ldots, \bar{x}_j), \qquad 0 \le j \le n.$$

Dowód Dowód przeprowadzimy podobnie jak dla węzłów jednokrotnych. Niech $w_{i,j} \in \Pi_{j-i}$ oznacza wielomian interpolacyjny Hermite'a oparty na (być może powtarzających się) węzłach $\bar{x}_i, \bar{x}_{i+1}, \ldots, \bar{x}_j$. To znaczy, $w_{i,j}$ interpoluje f w węzłach x_s takich, że x_s występuje w ciągu $\bar{x}_i, \ldots \bar{x}_j$, a jego krotność jest liczbą powtórzeń x_s w tym ciągu.

Zauważmy najpierw, że dla $\bar{x}_i \neq \bar{x}_j$ zachodzi znany nam już wzór (6.7),

$$w_{i,j}(x) = \frac{(x - \bar{x}_i) w_{i+1,j}(x) - (x - \bar{x}_j) w_{i,j-1}(x)}{\bar{x}_j - \bar{x}_i}. \tag{6.8}$$

Rzeczywiście, oznaczmy przez $v(x)$ prawą stronę powyższej równości. Dla k mniejszego od krotności danego węzła x_s w ciągu $\bar{x}_i, \ldots \bar{x}_j$, mamy $w_{i+1,j}^{(k-1)}(x_s) = w_{i,j-1}^{(k-1)}(x_s)$, a ponieważ

$$\begin{aligned} v^{(k)}(x) &= \frac{k\,(w_{i+1,j}^{(k-1)}(x) - w_{i,j-1}^{(k-1)}(x))}{\bar{x}_j - \bar{x}_i} \\ &\quad + \frac{(x - \bar{x}_i) w_{i+1,j}^{(k)}(x) - (x - \bar{x}_j) w_{i,j-1}^{(k)}(x)}{\bar{x}_j - \bar{x}_i}, \end{aligned}$$

to

$$v^{(k)}(x_s) = \frac{(x_s - \bar{x}_i) w_{i+1,j}^{(k)}(x_s) - (x_s - \bar{x}_j) w_{i,j-1}^{(k)}(x_s)}{\bar{x}_j - \bar{x}_i}.$$

Korzystając z tego wzoru sprawdzamy, że v spełnia odpowiednie warunki interpolacyjne, a stąd $w_{i,j} = v$.

Dalej postępujemy indukcyjnie ze względu na n. Dla $n = 0$ mamy $b_0 = f(x_0)$. Dla $n \geq 1$ wystarczy pokazać, że $b_n = f(\bar{x}_0, \bar{x}_1, \ldots, \bar{x}_n)$. W tym celu rozpatrzymy dwa przypadki.

Jeśli $\bar{x}_0 = \bar{x}_n$ to mamy jeden węzeł x_0 o krotności $n+1$. Wielomian interpolacyjny jest wtedy postaci

$$w_f(x) = \sum_{j=0}^{n} \frac{f^{(j)}(x_0)}{j!} (x - x_0)^j,$$

a stąd $b_n = f^{(n)}(x_0)/(n!) = f(\underbrace{x_0, \ldots, x_0}_{n+1})$. Jeśli zaś $\bar{x}_0 \neq \bar{x}_j$ to równość $b_n = f(\bar{x}_0, \bar{x}_1, \ldots, \bar{x}_n)$ wynika z (6.8) po podstawieniu $i = 0$ i $j = n$, oraz z założenia indukcyjnego. □

Uwagi i uzupełnienia

U. 6.1 Algorytm Hornera okazuje się optymalny. Każdy inny algorytm obliczający dokładną wartość wielomianu znając jego współczynniki wymaga wykonania co najmniej n mnożeń i n dodawań. Algorytm Hornera jest też numerycznie poprawny, zob. Ćw. 6.1.

U. 6.2 Mając obliczone współczynniki b_j wielomianu interpolacyjnego w postaci Newtona, można łatwo przejść do jego współczynników a_j w bazie potęgowej. Algorytm wynika bowiem z następującego wzoru rekurencyjnego.

Dla $i = n, n-1, \ldots, 0$, niech $a_j^{(i)}$, $i \le j \le n$, będą odpowiednimi współczynnikami w rozwinięciu

$$b_i + b_{i+1}(x - x_i) + \cdots + b_n(x - x_i)\cdots(x - x_{n-1}) \;=\; a_i^{(i)} + a_{i+1}^{(i)} x + \cdots + a_n^{(i)} x^{n-i}.$$

Wtedy $a_n^{(i)} = b_n$ oraz

$$a_j^{(i)} \;=\; a_j^{(i+1)} \;-\; x_i * a_{j+1}^{(i+1)}, \qquad i \le j \le n-1.$$

Uwzględniając fakt, że poszukiwanymi współczynnikami są $a_j = a_j^{(0)}$, $0 \le j \le n$, algorytm może być zrealizowany następująco:

for $j := 0$ **to** n **do** $a[j] := b[j]$;
for $i := n-1$ **downto** 0 **do**
for $j := i$ **to** $n-1$ **do**
  $a[j] := a[j] - x[i] * a[j+1]$.

U. 6.3 Okazuje się, że przy realizacji w fl_ν algorytmu różnic dzielonych istotną rolę odgrywa porządek węzłów. Można pokazać, że algorytm liczenia $f(t_0, \ldots, t_n)$ jest numerycznie poprawny ze względu na dane interpolacyjne $f^{(i)}(t_j)$, o ile węzły są uporządkowane nierosnąco lub niemalejąco, zob. Ćw. 6.5.

U. 6.4 Zauważmy, ze pojęcie różnicy dzielonej formalnie zdefiniowaliśmy jedynie dla ciągu węzłów postaci $x_0, \ldots, x_0, x_1, \ldots, x_1, \ldots, x_k, \ldots, x_k$, gdzie x_j są parami różne. Tą definicję można rozszerzyć do dowolnego ciągu węzłów. Można bowiem powiedzieć, że $f(t_0, t_1, \ldots, t_n)$ jest współczynnikiem przy x^n wielomianu $w_{t_0,\ldots,t_n} \in \Pi_n$ interpolującego f w węzłach t_j (uwzględniając krotności). Równoważnie,

$$f(t_0, t_1, \ldots, t_n) \;=\; \frac{w^{(n)}_{t_0,\ldots,t_n}}{n!}.$$

Ćwiczenia

Ćw. 6.1 Pokazać numeryczną poprawność algorytmu Hornera obliczania wartości wielomianu ze względu na dane współczynniki a_j tego wielomianu.

Ćw. 6.2 Pokazać, że algorytm Hornera obliczania wartości $w(\xi)$ wielomianu danego w postaci potęgowej jest jednocześnie algorytmem dzielenia tego wielomianu przez jednomian $(x-\xi)$. Dokładniej, jeśli $w(x)=\sum_{j=0}^n a_jx^j$ to

$$w(x) \;=\; \Big(\sum_{j=1}^n v_jx^{j-1}\Big)*(x-\xi)\;+\;v_0,$$

gdzie v_j są zdefiniowane tak jak w algorytmie Hornera.

Ćw. 6.3 Zaproponować możliwie tani algorytm obliczający współczynniki wielomianu w bazie Newtona, dysponując współczynnikami tego wielomianu w bazie potęgowej.

Ćw. 6.4 Pokazać, że dla różnych węzłów $t_0,\ldots,t_n$ mamy

$$f(t_0,t_1,\ldots,t_n) \;=\; \sum_{j=0}^n \frac{f(t_j)}{(t_j-t_0)\cdots(t_j-t_{j-1})(t_j-t_{j+1})\cdots(t_j-t_n)}.$$

Ćw. 6.5 Niech $t_0<t_1<\ldots<t_n$. Pokazać, że realizując w fl_ν algorytm różnic dzielonych, zamiast dokładnej wartości $f(t_0,t_1,\ldots,t_n)$ otrzymujemy $fl_\nu\Big(f(t_0,t_1,\ldots,t_n)\Big)$, która jest dokładną różnicą dzieloną dla danych $f(t_j)(1+\varepsilon_j)$, gdzie $|\varepsilon_j|\le K\nu$, $0\le j\le n$.
Wskazówka Rozpatrzyć najpierw proste przypadki $n=1,2$. Dla dowolnego n skorzystać z Ćw. 6.4.

Ćw. 6.6 Uzasadnić, że iloraz różnicowy jest funkcją symetryczną, tzn. dla dowolnej permutacji $t_{i_0},\ldots,t_{i_s}$ węzłów $t_0,\ldots,t_s$ mamy

$$f(t_{i_0},\ldots,t_{i_s}) \;=\; f(t_0,\ldots,t_s).$$

Ćw. 6.7 Niech w będzie wielomianem stopnia dokładnie n. Pokazać, że dla ustalonych $t_1,t_2,\ldots,t_s$ funkcja

$$w_{t_1,\ldots,t_s}(t) \;=\; w(t,t_1,t_2,\ldots,t_s), \quad t\in\mathbf{R},$$

jest wielomianem stopnia dokładnie $n-s$ dla $s\le n$, oraz jest zerem dla $s\ge n+1$.

Ćw. 6.8 Pokazać, że jeśli funkcja $f : D \to \mathbf{R}$ jest n-krotnie różniczkowalna na D w sposób ciągły, $f \in C^{(n)}(D)$, to

$$f(t_0, t_1, \ldots, t_{n-1}, t_0) = \lim_{t_n \to t_0} f(t_0, t_1, \ldots, t_{n-1}, t_n).$$

Ćw. 6.9 Stosując algorytm różnic dzielonym znaleźć wielomian w postaci Hermite'a interpolujący funkcję $f(x) = x^4$ w trzech dwukrotnych węzłach równych $0, 1$ i 2.

æ

Rozdział 7

Interpolacja a aproksymacja funkcji

Gdy mamy do czynienia z funkcją, która jest “skomplikowana” to często dobrze jest zastąpić ją funkcją “prostszą”. Mówimy wtedy o *aproksymacji (przybliżaniu) funkcji*. Funkcję musimy również aproksymać wtedy, gdy nie jesteśmy w stanie uzyskać pełnej o niej informacji. Na przykład, gdy funkcja reprezentuje pewien proces fizyczny to często zdarza się, że dysponujemy jedynie ciągiem próbek, czyli wartościami tej funkcji w pewnych punktach. Jasne jest, że chcielibyśmy przy tym, aby błąd aproksymacji był możliwie mały.

Z tego punktu widzenia, intepolacja wielomianowa może być traktowana jako jeden ze sposobów aproksymacji funkcji, opartym na próbkowaniu. Naturalnym staje się więc pytanie o błąd takiej aproksymacji.

7.1 Błąd interpolacji wielomianowej

Niech $x_0, x_1, \ldots, x_n$ będą (niekoniecznie różnymi) węzłami należącymi do pewnego (być może nieskończonego) przedziału $D \subset \mathbf{R}$. Dla danej funkcji $f : D \to \mathbf{R}$, przez w_f oznaczymy, tak jak w poprzednim rozdziale, wielomian interpolacyjny stopnia co najwyżej n interpolujący f w zadanych węzłach. W przypadku węzłów wielokrotnych jest to oczywiście wielomian interpolacyjny Hermite'a.

Lemat 7.1 *Dla dowolnego punktu $\bar{x} \in D$ błąd interpolacji w $\bar{x}$ wyraża się wzorem*

$$f(\bar{x}) - w_f(\bar{x}) \;=\; (\bar{x} - x_0)(\bar{x} - x_1)\cdots(\bar{x} - x_n) f(x_0, x_1, \ldots, x_n, \bar{x}). \quad (7.1)$$

Jeśli ponadto $f \in \mathbf{C}^{(n+1)}(D)$, czyli pochodna $f^{(n+1)}$ w D istnieje i jest ciągła, to

$$f(\bar{x}) - w_f(\bar{x}) \;=\; (\bar{x} - x_0)(\bar{x} - x_1)\cdots(\bar{x} - x_n) \frac{f^{(n+1)}(\xi)}{(n+1)!},$$

gdzie $\xi = \xi(\bar{x})$ jest pewnym punktem należącym do najmniejszego przedziału zawierającego punkty $x_0, x_1, \ldots, x_n, \bar{x}$.

Dowód Możemy założyć, że $\bar{x}$ nie jest żadnym z węzłów x_j, $0 \le j \le n$. Niech $\bar{w}_f \in \Pi_{n+1}$ będzie wielomianem interpolacyjnym funkcji f opartym na węzłach $x_0, \ldots, x_n$ i dodatkowo na węźle $\bar{x}$. Mamy wtedy

$$\bar{w}_f(x) \;=\; w_f(x) \;+\; (x - x_0)(x - x_1)\cdots(x - x_n) f(x_0, x_1, \ldots, x_n, \bar{x}),$$

a ponieważ z warunku interpolacyjnego $f(\bar{x}) = \bar{w}_f(\bar{x})$, to mamy też (7.1).

Aby pokazać drugą część lematu, rozpatrzmy funkcję $\psi : D \to \mathbf{R}$,

$$\begin{aligned} \psi(x) &= f(x) - \bar{w}_f(x) \\ &= f(x) - w_f(x) - (x - x_0)(x - x_1)\cdots(x - x_n) f(x_0, \ldots, x_n, \bar{x}). \end{aligned}$$

Z warunków interpolacyjnych na $\bar{w}_f \in \Pi_{n+1}$ wynika, że funkcja ψ ma punkty zerowe o łącznej krotności co najmniej $n+2$. Wykorzystując twierdzenie Rolle'a wnioskujemy stąd, że ψ' ma zera o łącznej krotności co najmniej $n+1$, ψ'' ma zera o łącznej krotności co najmniej n, itd. W końcu funkcja $\psi^{(n+1)}$ zeruje się w co najmniej jednym punkcie $\xi = \xi(\bar{x})$ należącym do najmniejszego przedziału zawierającego $x_0, x_1, \ldots, x_n, \bar{x}$. Wobec tego, że $w_f^{(n+1)} \equiv 0$, a $(n+1)$-sza pochodna wielomianu $(x - x_0)\cdots(x - x_n)$ wynosi $(n+1)!$, mamy

$$0 \;=\; \psi^{(n+1)}(\xi) \;=\; f^{(n+1)}(\xi) - (n+1)! f(x_0, \ldots, x_n, \bar{x}).$$

Stąd

$$f(x_0, x_1, \ldots, x_n, \bar{x}) \;=\; \frac{f^{(n+1)}(\xi)}{(n+1)!},$$

co kończy dowód. □

Zwykle interesuje nas nie tyle błąd w ustalonym punkcie $\bar{x} \in D$, ale na całym przedziale D. Zakładając teraz, że przedział D jest domknięty, czyli

$$D = [a,b]$$

dla pewnych $-\infty < a < b < +\infty$, błąd ten będziemy mierzyć w normie *jednostajnej* (Czebyszewa). Dla funkcji ciągłej $g : [a,b] \to \mathbf{R}$, norma ta jest zdefiniowana jako

$$\|g\|_{\mathbf{C}([a,b])} = \max_{x \in D} |g(x)|.$$

Niech $F^r_M([a,b])$, gdzie $r \geq 0$, będzie klasą funkcji

$$F^r_M([a,b]) = \{ f \in \mathbf{C}^{(r+1)}([a,b]) : \|f^{(r+1)}\|_{\mathbf{C}([a,b])} \leq M \},$$

gdzie $0 < M < \infty$. Mamy następujące twierdzenie.

Twierdzenie 7.1 *Załóżmy, że każdą funkcję $f \in F^r_M([a,b])$ aproksymujemy jej wielomianem interpolacyjnym $w_f \in \Pi_r$ opartym na $r+1$ węzłach $x_0, \ldots, x_r \in [a,b]$. Wtedy maksymalny błąd takiej aproksymacji wynosi*

$$\begin{aligned} e(F^r_M([a,b]); x_0, x_1, \ldots, x_r) &= \max_{f \in F^r_M([a,b])} \|f - w_f\|_{\mathbf{C}([a,b])} \\ &= \frac{M}{(r+1)!} \cdot \max_{a \leq x \leq b} |(x-x_0)\cdots(x-x_r)|. \end{aligned}$$

Dowód Oszacowanie górne wynika bezpośrednio z Lematu 7.1, bowiem dla $f \in F^r_M([a,b])$ mamy

$$\begin{aligned} \|f - w_f\|_{\mathbf{C}([a,b])} &= \max_{a \leq x \leq b} |f(x) - w_f(x)| \\ &= \max_{a \leq x \leq b} |(x-x_0)\cdots(x-x_r)| \frac{|f^{(r+1)}(\xi(x))|}{(r+1)!} \\ &\leq \frac{M}{(r+1)!} \max_{x \in D} |(x-x_0)\cdots(x-x_r)|. \end{aligned}$$

Z drugiej strony zauważmy, że dla wielomianu $v(x) = Mx^{r+1}/(r+1)!$ mamy $v \in F_M^r([a,b])$ oraz

$$\|v - w_v\|_{\mathbf{C}([a,b])} \;=\; \frac{M}{(r+1)!} \cdot \max_{a \le x \le b} |(x-x_0)\cdots(x-x_r)|,$$

co kończy dowód. $\square$

Zauważmy, że błąd aproksymacji $e(F_M^r([a,b]); x_0, \ldots, x_r)$ w istotny sposób zależy od wyboru węzłów x_j. Naturalne jest więc teraz następujące pytanie. W których punktach x_j przedziału $[a,b]$ należy obliczać wartości funkcji, aby błąd był minimalny? Problem ten sprowadza się oczywiście do minimalizacji wielkości $\max_{a \le x \le b} |(x-x_0)\cdots(x-x_r)|$ względem węzłów x_j.

Twierdzenie 7.2 *Błąd aproksymacji $F_M^r([a,b])(x_0, \cdots, x_r)$ jest minimalny gdy węzły*

$$x_j^* \;=\; \frac{b-a}{2} \cdot \cos\left(\frac{2j+1}{2r+2}\pi\right) \;+\; \frac{a+b}{2}, \qquad 0 \le j \le r.$$

Ponadto, dla węzłów optymalnych x_j^* mamy

$$e(F_M^r([a,b]); x_0^*, \ldots, x_r^*) \;=\; \frac{2M}{(r+1)!}\left(\frac{b-a}{4}\right)^{r+1}.$$

Dowód tego twierdzenia opiera się na własnościach pewnego ważnego ciągu wielomianów, który teraz przedstawimy.

7.2 Wielomiany Czebyszewa

Ciąg $\{T_k\}_{k \ge 0}$ *wielomianów Czebyszewa* (pierwszego rodzaju) zdefiniowany jest indukcyjnie jako

$$\begin{aligned} T_0(x) &= 1, \\ T_1(x) &= x, \\ T_{k+1}(x) &= 2xT_k(x) - T_{k-1}(x), \qquad \text{dla} \quad k \ge 1. \end{aligned}$$

Zauważmy, że T_k jest wielomianem stopnia dokładnie k o współczynniku przy x^k równym 2^{k-1} $(k \geq 1)$. Ponadto wielomian T_k można dla $|x| \leq 1$ przedstawić w postaci

$$T_k(x) \;=\; \cos(k \arccos x). \tag{7.2}$$

Rzeczywiście, łatwo sprawdzić, że jest to prawdą dla $k = 0, 1$. Stosując podstawienie $\cos t \;=\; x$, $0 \;\leq\; t \;\leq\; \pi$, oraz wzór na sumę cosinusów otrzymujemy dla $k \geq 1$

$$\cos((k+1)t) \;=\; 2 \cdot \cos t \cos(kt) \;-\; \cos((k-1)t),$$

co jest równoważne formule rekurencyjnej dla T_{k+1}.

Ze wzoru (7.2) wynikają również inne ważne własności wielomianów Czebyszewa. Norma wielomianu Czebyszewa na $[-1, 1]$ wynosi

$$\|T_k\|_{\mathbf{C}([-1,1])} \;=\; \max_{-1 \leq x \leq 1} |T_k(x)| \;=\; 1$$

i jest osiągana w $k + 1$ punktach tego przedziału równych

$$y_j \;=\; \cos\Big(\frac{j}{k}\pi\Big), \qquad 0 \leq j \leq k, \tag{7.3}$$

przy czym $T_k(y_j) = (-1)^j$.

W końcu, k-ty wielomian Czebyszewa T_k ma dokładnie k pojedynczych zer w $[-1, 1]$ równych

$$z_j \;=\; \cos\Big(\frac{2j+1}{2r}\pi\Big), \qquad 0 \leq j \leq k-1.$$

Konsekwencją wymienionych własności jest następujący lemat, który jest kluczem do dowodu Twierdzenia 7.2.

Przez $\overline{\Pi}_k$ oznaczymy klasę wielomianów stopnia k o współczynniku wiodącym równym 1, tzn.

$$\overline{\Pi}_k \;=\; \{\, w \in \Pi_k \,:\, w(x) = x^k + \cdots \}.$$

Lemat 7.2 *Niech $k \geq 1$. W klasie $\overline{\Pi}_k$ minimalną normę jednostajną na przedziale $[-1, 1]$ ma wielomian $w^* = 2^{1-k} T_k$, tzn.*

$$\min_{w \in \overline{\Pi}_k} \|w\|_{\mathbf{C}([-1,1])} \;=\; \|w^*\|_{\mathbf{C}([-1,1])} \;=\; \frac{1}{2^{k-1}}.$$

Dowód Zauważmy najpierw, że $w^* \in \overline{\Pi}_k$ oraz $\|w^*\|_{\mathbf{C}([-1,1])} = 2^{1-k}$. Wystarczy więc pokazać, że norma każdego wielomianu z $\overline{\Pi}_k$ jest nie mniejsza niż 2^{1-k}.

Załóżmy, że jest przeciwnie, tzn. istnieje wielomian $w \in \overline{\Pi}_k$ taki, że

$$\|w\|_{\mathbf{C}([-1,1])} < \frac{1}{2^{k-1}} = \|w^*\|_{\mathbf{C}([-1,1])}.$$

Rozpatrzmy funkcję $\psi = w^* - w$. Ponieważ dla punktów "maksymalnych" zdefiniowanych w (7.3) mamy $w^*(y_{k-j}) = (-1)^j 2^{1-k}$ oraz $|w(y_{k-j})| < 2^{1-k}$, to

$$\psi(y_{k-j}) \begin{cases} > 0 & j\text{-parzyste,} \\ < 0 & j\text{-nieparzyste.} \end{cases}$$

$0 \le j \le k$. Stąd ψ ma co najmniej jedno zero w każdym z przedziałów (y_i, y_{i+1}) dla $0 \le i \le k-1$, czyli w sumie k zer. Z drugiej strony, ψ jest wielomianem stopnia co najwyżej $k-1$ (bo współczynniki przy x^k w wielomianach w^* i w redukują się), a więc $\psi = 0$ i $w^* = w$. □

Dowód Twierdzenia 7.2 wynika teraz bezpośrednio z Lematu 7.2. Zauważmy bowiem, że wielomian $(x-x_0)(x-x_1)\cdots(x-x_r)$ jest w klasie $\overline{\Pi}_{r+1}$. Stąd dla $[a,b] = [-1,1]$ optymalnymi węzłami są zera z_j wielomianu Czebyszewa, przy których

$$(x-z_0)(x-z_1)\cdots(x-z_r) = \frac{T_{r+1}(x)}{2^r}.$$

Jeśli przedział $[a,b]$ jest inny niż $[-1,1]$, należy dokonać liniowej zamiany zmiennych tak, aby przeszedł on na $[-1,1]$. Bezpośrednie sprawdzenie pokazuje, że w klasie $\overline{\Pi}_{r+1}$ minimalną normę Czebyszewa na przedziale $[a,b]$ ma wielomian

$$w^*_{a,b}(x) = \left(\frac{b-a}{2}\right)^{r+1} w^*\left(\frac{2x-(a+b)}{b-a}\right).$$

Stąd

$$\|w^*_{a,b}\|_{\mathbf{C}([a,b])} = \left(\frac{b-a}{2}\right)^{r+1}\frac{1}{2^r} = 2\left(\frac{b-a}{4}\right)^{r+1}$$

i węzły x^*_j są optymalne. □

7.3 Interpolacja kawałkami wielomianowa

W praktyce interesuje nas konstrukcja nie jednej, a raczej ciągu aproksymacji, które przybliżałyby daną funkcję f z coraz większą dokładnością. Cel ten można spróbować osiągnąc przez wzięcie jako n-tej aproksymacji wielomianu interpolacyjnego $w_{f,n} \in \Pi_n$ opartego na pewnych węzłach x_j, $0 \le j \le n$. Rzeczywiście, jeśli f jest na $[a,b]$ nieskończenie wiele razy różniczkowalna to n-ty błąd dla dowolnego układu węzłów jest ograniczony przez

$$\|f - w_{f,n}\|_{\mathbf{C}([a,b])} \le \frac{\|f^{(n+1)}\|_{\mathbf{C}([a,b])}}{(n+1)!}(b-a)^{n+1}, \qquad (7.4)$$

a więc szybko dąży zera wraz z $n \to \infty$, gdy np. wszystkie pochodne funkcji f są wspólnie ograniczone przez pewną stałą. Tak jest np. dla $f(x) = e^x$. Jeszcze lepsze oszacowanie dostaniemy, gdy wybierzemy węzły Czebyszewa.

Podejście to ma jednak pewne wady. Zwiększając stopień wielomianu interpolacyjnego sprawiamy, że aproksymacja, a także obliczenie współczynników wielomianu interpolacyjnego, stają się coraz bardziej skomplikowane, a tego chcielibyśmy uniknąć. Poza tym, założenie o istnieniu i wspólnej ograniczoności wszystkich pochodnych funkcji nie zawsze jest spełnione (np. w rozpatrywanej już klasie $F_M^r([a,b])$), co powoduje, że oszacowanie (7.4) nie zawsze jest prawdziwe, a nawet błąd $\|f - w_{f,n}\|_{\mathbf{C}([a,b])}$ może nie maleć do zera, zob. U. 7.1.

Podamy teraz inny sposób konstrukcji ciągu aproksymacji, który zapewnia zbieżność w klasie $F_M^r([a,b])$, ale nie wymaga nadmiernego zwiększania stopnia wielomianu interpolacyjnego.

Postąpimy zgodnie ze znaną zasadą "dziel i rządź". Podzielimy przedział $[a,b]$ na k równych podprzedziałów $[t_{i-1}, t_i]$, $1 \le i \le k$, i zastosujemy na każdym z nich interpolację wielomianową stopnia r. Dokładniej, dla każdego i wybieramy (być może wielokrotne) węzły $x_{i,j} \in [t_{i-1}.t_i]$, $0 \le j \le r$, po czym aproksymujemy f na $[t_{i-1}, t_i]$ jej wielomianem interpolacyjnym opartym na tych węzłach. Tak skonstruowaną kawałkami wielomianową aproksymację funkcji $f : [a,b] \to \mathbf{R}$

oznaczymy przez $\bar{w}_f$. Zauważmy, że $\bar{w}_f$ wykorzystuje na całym $[a,b]$ węzły o łącznej krotności n spełniającej nierówność

$$n \le k(r+1). \tag{7.5}$$

Twierdzenie 7.3 *Błąd aproksymacji kawałkami wielomianowej w klasie funkcji $F_M^r([a,b])$ jest ograniczony przez*

$$\max_{f \in F_M^r([a,b])} \|f - \bar{w}_f\|_{\mathbf{C}([a,b])} \le \frac{M}{(r+1)!}\Big(\frac{b-a}{k}\Big)^{r+1} \le C \cdot \Big(\frac{1}{n}\Big)^{r+1},$$

gdzie

$$C = \frac{M(b-a)^{r+1}(r+1)^{r+1}}{(r+1)!}.$$

Jeśli ponadto węzły $x_{i,j}$, $0 \le j \le r$, są na każdym przedziale $[t_{i-1}, t_i]$ węzłami Czebyszewa, to dla odpowiedniej aproksymacji $\bar{w}_f^*$ mamy

$$\max_{f \in F_M^r([a,b])} \|f - \bar{w}_f^*\|_{\mathbf{C}([a,b])} = \frac{2M}{(r+1)!}\Big(\frac{b-a}{4k}\Big)^{r+1} = C^* \cdot \Big(\frac{1}{n}\Big)^{r+1},$$

gdzie

$$C^* = \Big(\frac{1}{2}\Big)^{2r+1} C = \frac{2M(b-a)^{r+1}(r+1)^{r+1}}{4^{r+1}(r+1)!}.$$

Dowód Niech $f \in F_M^r([a,b])$ i $x \in [t_{i-1}, t_i]$. Z Lematu 7.1 wynika, że

$$|f(x) - \bar{w}_f(x)| = \frac{|f^{(r+1)}(\xi(x))|}{(r+1)!}|(x - x_{i,0})(x - x_{i,1}) \cdots (x - x_{i,r})|,$$

a ponieważ $|x - x_{i,j}| \le t_i - t_{i-1} = (b-a)/k$, to

$$|f(x) - \bar{w}_f(x)| \le \frac{M}{(r+1)!}\Big(\frac{b-a}{k}\Big)^{r+1}.$$

Ta nierówność wraz z (7.5) dowodzi pierwszą część twierdzenia.

Dla dowodu drugiej części wystarczy wykorzystać Twierdzenie 7.2, aby zauważyć, że

$$\|f - \bar{w}_f^*\|_{\mathbf{C}([a,b])} = \max_{a \le x \le b} |f(x) - \bar{w}_f^*(x)| \le \frac{2M}{(r+1)!}\Big(\frac{b-a}{4k}\Big)^{r+1}$$

oraz, że dla $f(x) = Mx^{r+1}/(r+1)!$ mamy powyżej równość. $\square$

Otrzymaliśmy, że interplacja kawałkami wielomianami stopnia r korzystająca z n wartości funkcji daje w klasie funkcji $F_M^r([a,b])$ błąd rzędu $n^{-(r+1)}$. Błąd ten zbiega więc do zera, gdy liczba n wartości funkcji rośnie do nieskończoności, przy czym charakter zbieżności (tzn. przy zaniedbaniu stałej przy $n^{-(r+1)}$) nie zależy ani od długości przedziału $[a,b]$, ani od doboru "wewnętrznyh" węzłów $x_{i,j}$, $0 \le j \le r$.

Można pokazać, że powyższa aproksymacja jest w klasie $F_M^r([a,b])$ optymalna, tzn. błąd dowolnej innej metody aproksymacji korzystającej z wartości funkcji w n punktach dąży do zera nie szybciej niż $n^{-(r+1)}$; zob. U.7.2.

Uwagi i uzupełnienia

U. 7.1 Niech $f(x) = (1+x^2)^{-1}$. Okazuje się, że jeśli $w_{f,n}$ jest wielomianem interpolacyjnym dla f opartym na węzłach równoodległych przedziału $[-5,5]$, tzn. $x_j = -5 + 10j/n$, $0 \le j \le n$, to ciąg $w_{f,n}(x)$ jest zbieżny do $f(x)$ dla $|x| < 3.63...$ i rozbieżny dla $|x| > 3.63....$ To znaczy, błąd interpolacji funkcji f na całym przedziale $[-5,5]$ nie maleje do zera, gdy stopień wielomianu interpolacyjnego rośnie do ∞.

U. 7.2 Pokażemy teraz, że aproksymacja funkcjami kawałkami wielomianowymi jest optymalna w $F_M^r([a,b])$ ze względu na rząd zbieżności błędu. Dokładniej pokażemy, że dla dowolnej aproksymacji $A(f)$ funkcji f wykorzystującej wartości f lub jej pochodnych w co najwyżej n różnych punktach, tzn. dla A postaci

$$A(f) = \phi(f(x_1),\ldots,f^{(r+1)}(x_1),\ldots,f(x_n),\ldots,f^{(r+1)}(x_n))$$

mamy

$$\sup_{f\in F_M^r([a,b])} \|f - A(f)\|_{\mathbf{C}([a,b])} \ge C_1\, M\, n^{-(r+1)}, \tag{7.6}$$

gdzie $C_1 > 0$ jest pewną stałą niezależną od A, M i n.

W tym celu wybierzmy dowolną niezerową funkcję $\psi : \mathbf{R} \to \mathbf{R}$ spełniającą warunki:

1. $\psi \in F_1^r(\mathbf{R})$,

2. $\psi(x) = 0$ dla $x \notin [0,1]$.

Niech s będzie takim indeksem, że $h_s = x_{s+1} - x_s \geq (b-a)/(n+1)$. Wtedy dla funkcji ψ_i, $i = 1, 2$, zdefiniowanych jako

$$\psi_1(x) \;=\; -\psi_2(x) \;=\; M h_s^{r+1} \psi\Big(\frac{x - x_s}{h_s}\Big)$$

mamy $\psi_i \in F_M^r([a,b])$ oraz informacja o ψ_i jest zerowa, $\psi_i^{(p)}(x_j) = 0$ dla $0 \leq p \leq r+1$, $1 \leq j \leq n$. Stąd $\phi_0 = \phi(\underbrace{0, \ldots, 0}_{n(r+2)})$ jest aproksymacją dla obu funkcji ψ_i i z nierówności trójkąta dla norm dostajemy

$$\begin{aligned} &\max\{\|\psi_1 - \phi_0\|_{\mathbf{C}([a,b])}, \|\psi_2 - \phi_0\|_{\mathbf{C}([a,b])}\} \\ &\quad\geq\; \|\psi_1 - \psi_2\|_{\mathbf{C}([a,b])}/2 \;=\; \|\psi_1\|_{\mathbf{C}([a,b])} \\ &\quad=\; M h_s^{r+1} \|\psi\|_{\mathbf{C}([0,1])} \;\geq\; M\Big(\frac{b-a}{n+1}\Big)^{r+1} \|\psi\|_{\mathbf{C}([0.1])}. \end{aligned}$$

Nierówność (7.6) zachodzi więc z $C_1 = (2(b-a))^{r+1}\|\psi\|_{\mathbf{C}([0.1])}$. (Zob. też Ćw. 8.3.)

U. 7.3 Załóżmy, że funkcje z klasy $F_M^r([a,b])$ chcemy aproksymować z dokładnością $\varepsilon > 0$ w normie Czebyszewa używając możliwie mało obliczeń wartości funkcji. Z Twierdzenia 7.3 i U. 7.2 wynika, że musimy wtedy obliczać co najmniej rzędu $(1/\varepsilon)^{r+1}$ wartości funkcji. Co więcej, aproksymacja kawałkami wielomianowa daje błąd ε obliczając właśnie tyle wartości funkcji. Mówimy, że *złożoność informacyjna* zadania jest rzędu $(1/\varepsilon)^{r+1}$.

Ćwiczenia

Ćw. 7.1 Niech dane będą $\varepsilon > 0$ i funkcja $f : [a,b] \to \mathbf{R}$, która jest nieskończenie wiele razy różniczkowalna i wszystkie jej pochodne są na $[a,b]$ ograniczone przez M. Napisać program znajdujący stopień n oraz współczynniki w bazie Newtona wielomianu $w_{f,n} \in \Pi_n$ interpolacyjnego f z błędem

$$\|f - w_{f,n}\|_{\mathbf{C}([a,b])} \;\leq\; \varepsilon.$$

Rozpatrzyć dwa przypadki: gdy węzły interpolacji są równoodległe, oraz gdy węzły są czebyszewowskie.

Ćw. 7.2 Pokazać, że dla $k \geq 1$ mamy

$$T_k(x) \;=\; \frac{(x+\sqrt{x^2-1})^k + (x-\sqrt{x^2-1})^k}{2},$$

gdzie T_k jest k-tym wielomianem Czebyszewa.

Ćw. 7.3 Pokazać, że wielomiany Czebyszewa tworzą układ ortogonalny w przestrzeni $L^2_\rho(-1,1)$, gdzie waga $\rho(x) = (1-x^2)^{-1/2}$, tzn. iloczyn skalarny

$$\langle T_i, T_j\rangle \;=\; \int_{-1}^{1} \frac{T_i(x)T_j(x)}{\sqrt{1-x^2}}\,dx \;=\; \begin{cases} \pi & i=j=0, \\ \pi/2 & i=j\neq 0, \\ 0 & i\neq j. \end{cases}$$

Ćw. 7.4 Wielomiany Czebyszewa T_k, $0 \leq k \leq n$, tworzą bazę w przestrzeni Π_n. Podać algorytm obliczający współczynniki danego wielomianu w bazie Czebyszewa na podstawie jesgo współczynników w bazie potęgowej, albo w bazie Newtona.

Ćw. 7.5 Podać algorytm typu algorytmu Hornera obliczający wartość wielomianu $w(x)$ danego przez jego współczynniki w bazie Czebyszewa $\{T_k\}_{k\geq 0}$.

Ćw. 7.6 Niech w będzie wielomianem, którego rozwinięciem w bazie Czebyszewa jest $w(x) = \sum_{k=0}^{n} a_k T_k(x)$. Pokazać, że wielomian

$$w_1(x) \;=\; \sum_{k=0}^{n-1} a_k T_k(x)$$

najlepiej przybliża w w normie Czebyszewa na przedziale $[-1,1]$ spośród wszystkich wielomianów stopnia co najwyżej $n-1$, oraz

$$\|w - w_1\|_{\mathbf{C}([-1,1])} \;=\; |a_n|.$$

Czy $w_2(x) = \sum_{k=0}^{n-2} a_k T_k(x)$ najlepiej przybliża w w tej samej normie w klasie Π_{n-2}?

Ćw. 7.7 Załóżmy, że wielomian $w(x) = \sum_{k=0}^{n} a_k T_k(x)$ aproksymujemy wielomianem $w_i(x) = \sum_{k=0}^{n-i} a_k T_k(x)$. Uzasadnić, że

$$\|w - w_i\|_{\mathbf{C}([-1,1])} \;\leq\; \sum_{k=n-i+1}^{n} |a_k|.$$

Kiedy w tej nierówności mamy równość?

Ćw. 7.8 Niech dany będzie przedział skończony $[a, b]$ nie zawierający zera, oraz liczba c. Niech

$$\tilde{\Pi}_k \;=\; \{\, w \in \Pi_k : \, w(0) = c \,\}.$$

Pokazać, że w klasie $\tilde{\Pi}_k$ najmniejszą normę Czebyszewa na $[a, b]$ ma wielomian

$$w(x) \;=\; c \cdot \frac{T_k\left(\frac{2x-(a+b)}{b-a}\right)}{T_k\left(\frac{a+b}{a-b}\right)}$$

i wynosi ona

$$\|w\|_{\mathbf{C}([a,b])} \;=\; \left|\frac{c}{T_k\left(\frac{a+b}{a-b}\right)}\right|.$$

Jakie jest rozwiązanie, gdy $0 \in [a, b]$?

Rozdział 8

Interpolacja funkcjami sklejanymi

Interpolacja kawałkami wielomianami interpolacyjnymi daje mały błąd aproksymacji, ma jednak również pewne wady. Zauważmy, że w punktach t_j podziału przedziału $[a, b]$ na podprzedziały funkcja aproksymująca $\bar{w}_f$ nie jest w ogólności ciągła. W najlepszym wypadku, nawet gdy węzły $x_{i,j}$ na przedziale $[t_{i-1}, t_i]$ wybierzemy tak, że $x_{i,0} = t_{i-1}$ i $x_{i,r} = t_i$, $\forall i$, to $\bar{w}_f(x)$ co prawda będzie ciągła, ale nie będzie różniczkowalna w x_i. Ta własność jest niepożądana w przypadku gdy wiemy, że funkcja którą aproksymujemy jest "gładka". Wtedy lepiej jest zastosować innego rodzaju interpolację, np. posługując się funkcjami sklejanymi.

8.1 Co to są funkcje sklejane?

W ogólności przez funkcję sklejaną rozumie się każdą funkcję przedziałami wielomianową. Nas będą jednak interesować szczególne funkcje tego typu i dlatego termin *funkcje sklejane* zarezerwujemy dla funkcji przedziałami wielomianowych i posiadających dodatkowe własności, które teraz określimy.

Niech dany będzie przedział skończony $[a, b]$ i węzły

$$a = x_0 < x_1 < \cdots < x_n = b,$$

przy czym $n \geq 1$.

Definicja 8.1 *Funkcję* $s : \mathbf{R} \to \mathbf{R}$ *nazywamy funkcją sklejaną rzędu* r $(r \geq 1)$ *odpowiadającą węzłom* x_j, $0 \leq j \leq n$, *jeśli spełnione są następujące dwa warunki:*

(i) s *jest wielomianem stopnia co najwyżej* $2r-1$ *na każdym z przedziałów* $[x_{j-1}, x_j]$, *tzn.* $s_{|[x_{j-1},x_j]} \in \Pi_{2r-1}$, $1 \leq j \leq n$,

(ii) s *jest* $(2r-2)$*-krotnie różniczkowalna w sposób ciągły na całej prostej, tzn.* $s \in \mathbf{C}^{(2r-2)}(\mathbf{R})$.

Jeśli ponadto

(iii) s *jest wielomianem stopnia co najwyżej* $r-1$ *poza* (a,b), *tzn.* $s_{|(-\infty,a]}, s_{|[b,+\infty)} \in \Pi_{r-1}$,

to s *jest naturalną funkcją sklejaną.*

Klasę naturalnych funkcji sklejanych rzędu r opartych na węzłach x_j będziemy oznaczać przez $\mathcal{S}_r(x_0, \ldots, x_n)$, albo po prostu $\mathcal{S}_r$, jeśli węzły są ustalone.

Na przykład, funkcją sklejaną rzędu pierwszego $(r = 1)$ jest funkcja ciągła i liniowa na poszczególnych przedziałach $[x_{j-1}, x_j]$. Jest ona naturalna, gdy poza (a,b) jest funkcją stała. Tego typu funkcje nazywamy *liniowymi funkcjami sklejanymi.*

Najważniejszymi a praktycznego punktu widzenia są jednak funkcje sklejane rzędu drugiego odpowiadające $r = 2$. Są to funkcje, które są na $\mathbf{R}$ dwa razy różniczkowalne w sposób ciągły, a na każdym z podprzedziałów są wielomianami stopnia co najwyżej trzeciego. W tym przypadku mówimy o *kubicznych funkcjach sklejanych.* Funkcja sklejana kubiczna jest naturalna, gdy poza (a,b) jest wielomianem liniowym.

8.2 Interpolacja i gładkość

Pokażemy najpierw ważny lemat, który okaże się kluczem do dowodu dalszych twierdzeń.

Niech $W^r(a,b)$ będzie klasą funkcji $f : [a,b] \to \mathbf{R}$ takich, że f jest $(r-1)$ razy różniczkowalna na $[a,b]$ w sposób ciągły oraz $f^{(r)}(x)$ istnieje prawie wszędzie na $[a,b]$ i jest całkowalna z kwadratem, tzn.

$$\begin{aligned} W^r(a,b) &= \{\, f \in \mathbf{C}^{(r-1)}([a,b]) : \ f^{(r)}(x) \text{ istnieje p.w. na } [a,b] \\ &\qquad\qquad \text{oraz } f^{(r)} \in \mathcal{L}_2(a,b)\,\}. \end{aligned}$$

Oczywiście każda funkcja sklejana rzędu r (niekoniecznie naturalna) należy do $W^r(a,b)$.

Lemat 8.1 *Niech $f \in W^r(a,b)$ będzie funkcją zerującą się w węzłach, tzn.*

$$f(x_j) = 0, \qquad 0 \le j \le n.$$

Wtedy dla dowolnej naturalnej funkcji sklejanej $s \in \mathcal{S}_r$ mamy

$$\int_a^b f^{(r)}(x)s^{(r)}(x)\,dx = 0.$$

Dowód Dla $r=1$ funkcja s' jest przedziałami stała. Oznaczając przez a_j jej wartość na $[x_{j-1},x_j]$ dostajemy

$$\begin{aligned} \int_a^b f'(x)s'(x)\,dx &= \sum_{j=1}^n \int_{t_{j-1}}^{t_j} f'(x)a_j\,dx \\ &= \sum_{j=1}^n a_j(f(x_j)-f(x_{j-1})) = 0, \end{aligned}$$

ponieważ f zeruje się w t_j.

Rozpatrzmy teraz przypadek $r \ge 2$. Ponieważ

$$(f^{(r-1)}s^{(r)})' = f^{(r)}s^{(r)} + f^{(r-1)}s^{(r+1)},$$

to

$$\int_a^b f^{(r)}(x)s^{(r)}(x)\,dx = f^{(r-1)}(x)s^{(r)}(x)\Big|_a^b - \int_a^b f^{(r-1)}(x)s^{(r+1)}(x)\,dx.$$

Wobec tego, że s jest poza przedziałem (a,b) wielomianem stopnia co najwyżej $r-1$ oraz $s^{(r)}$ jest ciągła na $\mathbf{R}$, mamy $s^{(r)}(a) = 0 = s^{(r)}(b)$, a stąd

$$f^{(r-1)}(x)s^{(r)}(x)\Big|_a^b = 0.$$

Postępując podobnie, tzn. całkując przez części $r-1$ razy otrzymujemy w końcu

$$\int_a^b f^{(r)}(x)s^{(r)}(x)\,dx \;=\; (-1)^{r-1}\int_a^b f'(x)s^{(2r-1)}(x)\,dx.$$

Funkcja $s^{(2r-1)}$ jest przedziałami stała, a więc możemy teraz zastosować ten sam argument jak dla $r = 1$, aby pokazać, że ostatnia całka jest równa zeru. □

Funkcje sklejane chcielibyśmy zastosować do interpolacji funkcji. Ważne jest więc, aby odpowiednie zadanie interpolacyjne miało jednoznaczne rozwiązanie.

Twierdzenie 8.1 *Jeśli $n+1 \geq r$, to dla dowolnej funkcji $f : [a,b] \to \mathbf{R}$ istnieje dokładnie jedna naturalna funkcja sklejana $s_f \in \mathcal{S}_r$ interpolująca f w węzłach x_j, tzn. taka, że*

$$s_f(x_j) \;=\; f(x_j), \qquad 0 \leq j \leq n.$$

Dowód Pokażemy najpierw, że jedyną naturalną funkcją sklejaną interpolującą dane zerowe jest funkcja zerowa. Rzeczywiście, jeśli $s(x_j) = 0$ dla $0 \leq j \leq n$, to podstawiając w Lemacie 8.1 $f = s$ otrzymujemy

$$\int_a^b \left(s^{(r)}(x)\right)^2 dx \;=\; 0.$$

Stąd $s^{(r)}$ jest funkcją zerową, a więc s jest wielomianem stopnia co najwyżej $r-1$ zerującym się w co najmniej $n+1$ punktach x_j. Wobec tego, że $n+1 > r-1$, otrzymujemy $s \equiv 0$.

Zauważmy teraz, że problem znalezienia naturalnej funkcji sklejanej s interpolującej f można sprowadzić do rozwiązania kwadratowego układu równań liniowych. Na każdym przedziale $[x_{i-1}, x_i]$, $1 \leq i \leq n$, jest ona postaci

$$s(x) \;=\; w_i(x) \;=\; \sum_{j=0}^{2r-1} a_{i,j}x^j,$$

dla pewnych współczynników $a_{i,j} \in \mathbf{R}$, a na $(-\infty, a]$ i $[b, \infty)$ mamy odpowiednio

$$s(x) \;=\; w_0(x) \;=\; \sum_{j=0}^{r-1} a_{0,j}x^j$$

i

$$s(x) \;=\; w_{n+1}(x) \;=\; \sum_{j=0}^{r-1} a_{n+1,j} x^j.$$

Aby wyznaczyć s, musimy więc znaleźć ogółem $2r(n+1)$ współczynników $a_{i,j}$, przy czym są one związane $(2r-1)(n+1)$ warunkami jednorodnymi wynikającymi z gładkości,

$$w_i^{(k)}(x_i) \;=\; w_{i+1}^{(k)}(x_i)$$

dla $0 \le i \le n$ i $0 \le k \le 2r-2$, oraz $n+1$ niejednorodnymi warunkami interpolacyjnymi,

$$w_i(x_i) \;=\; f(x_i)$$

dla $0 \le i \le n$. Otrzymujemy więc układ $2r(n+1)$ równań liniowych ze względu na $2r(n+1)$ niewiadomych $a_{i,j}$.

Naturalna funkcja sklejana interpolująca f jest wyznaczona jednoznacznie wtedy i tylko wtedy, gdy układ ten ma jednoznaczne rozwiązanie. To zaś zachodzi, gdy zero jest jedynym rozwiązaniem układu jednorodnego. Rzeczywiście, układ jednorodny odpowiada zerowym warunkom interpolacyjnym, przy których, jak pokazaliśmy wcześniej, zerowa funkcja sklejana (której odpowiada $a_{i,j} = 0$, $\forall i, j$) jest jedynym rozwiązaniem zadania interpolacyjnego. □

Naturalne funkcje sklejane możemy więc używać do interpolacji funkcji. Pokażemy teraz inną ich własność, która jest powodem dużego praktycznego zainteresowania funkcjami sklejanymi.

Twierdzenie 8.2 *Niech $f \in W^r(a,b)$ i niech $s_f \in \mathcal{S}_r$ będzie naturalną funkcją sklejaną rzędu r interpolującą f w węzłach x_j, $0 \le j \le n$. Wtedy*

$$\int_a^b \left(f^{(r)}(x)\right)^2 dx \;\ge\; \int_a^b \left(s_f^{(r)}(x)\right)^2 dx. \tag{8.1}$$

Dowód Jeśli przedstawimy f w postaci $f = s_f + (f - s_f)$, to

$$\int_a^b \left(f^{(r)}(x)\right)^2 dx \;=\; \int_a^b \left(s_f^{(r)}(x)\right)^2 dx \,+\, \int_a^b \left((f-s_f)^{(r)}(x)\right)^2 dx$$
$$2\int_a^b s_f^{(r)}(x)(f-s_f)^{(r)}(x)\,dx.$$

Funkcja $f - s_f$ jest w klasie $W^r(a,b)$ i zeruje się w węzłach x_j, $0 \le j \le n$. Z Lematu 8.1 wynika więc, że $\int_a^b s_f^{(r)}(x)(f-s_f)^{(r)}(x)dx = 0$, a stąd (8.1). □

Wartość całki $\int_a^b (f^{(r)}(x))^2 dx$ może być w ogólności uważana za miarę gładkości funkcji. Nierówność (8.1) możemy więc zinterpretować w następujący sposób. Naturalna funkcja sklejana jest w klasie $W^r(a,b)$ najgładszą funkcją spełniającą dane warunki interpolacyjne w wybranych węzłach x_j.

Konsekwencją Lematu 8.1 są również inne aproksymacyjne własności naturalnych funkcji sklejanych, zob. U. 8.2.

Jak już wspomnieliśmy, najczęściej używanymi są kubiczne funkcje sklejane. Dlatego rozpatrzymy je oddzielnie.

8.3 Kubiczne funkcje sklejane

Jeśli zdecydowaliśmy się na użycie kubicznych funkcji sklejanych to powstaje problem wyznaczenia $s_f \in \mathcal{S}_2$ interpolującej daną funkcję f, tzn. takiej, że $s_f(x_i) = f(x_i)$ dla $0 \le i \le n$. W tym celu, na każdym przedziale $[x_i, x_{i+1}]$ przedstawimy s_f w postaci jej rozwinięcia w szereg Taylora w punkcie x_i,

$$s_f(x) \;=\; w_i(x) \;=\; a_i + b_i(x-x_i) + c_i\frac{(x-x_i)^2}{2} + d_i\frac{(x-x_i)^3}{6},$$

i podamy algorytm obliczania a_i, b_i, c_i, d_i dla $0 \le i \le n-1$.

Warunki brzegowe i warunki ciągłości dla s_f'' dają nam $w_0''(x_0) = 0 = w_{n-1}''(x_n)$ oraz $w_i''(x_{i+1}) = w_{i+1}''(x_{i+1})$, czyli

$$\begin{aligned} c_0 &= 0, \\ c_i + d_i h_i &= c_{i+1}, \qquad 0 \le i \le n-2, \\ c_{n-1} + d_{n-1}h_{n-1} &= 0, \end{aligned}$$

gdzie $h_i = x_{i+1} - x_i$. Stąd, przyjmując dodatkowo $c_n = 0$, otrzymujemy

$$d_i \;=\; \frac{c_{i+1} - c_i}{h_i}, \qquad 1 \le i \le n-1. \tag{8.2}$$

Z warunków ciągłości dla s_f' dostajemy z kolei

$$b_i + c_i h_i + d_i h_i^2/2 \;=\; b_{i+1}, \qquad 0 \le i \le n-2,$$

i uwzględniając (8.2)

$$b_{i+1} \;=\; b_i + h_i(c_{i+1} + c_i)/2, \qquad 0 \le i \le n-2. \tag{8.3}$$

Warunki ciągłości s_f dają w końcu

$$a_i + b_i h_i + c_i h_i^2/2 + d_i h_i^3/6 \;=\; a_{i+1}, \qquad 0 \le i \le n-2. \tag{8.4}$$

Równania (8.2),(8.3) i (8.4) definiują nam na odcinku $[a, b]$ naturalną kubiczną funkcję sklejaną. Ponieważ poszukiwana funkcja sklejana s_f ma interpolować f, mamy dotatkowych $n+1$ warunków interpolacyjnych $w_i(x_i) = f(x_i)$, $0 \le i \le n-1$, oraz $w_{n-1}(x_n) = f(x_n)$, z których

$$a_i \;=\; f(x_i), \qquad 0 \le i \le n-1. \tag{8.5}$$

Teraz możemy (8.4) przepisać jako

$$f(x_{i+1}) \;=\; f(x_i) + b_i h_i + c_i h_i^2 + d_i h_i^3/6,$$

przy czym wzór ten zachodzi również dla $i = n-1$. Po wyrugowaniu b_i i podstawieniu d_i z (8.2), mamy

$$b_i \;=\; f(x_i, x_{i+1}) + h_i(c_{i+1} + 2c_i)/6, \qquad 0 \le i \le n-1, \tag{8.6}$$

gdzie $f(x_i, x_{i+1})$ jest odpowiednią różnicą dzieloną. Możemy teraz powyższe wyrażenie na b_i podstawić w równaniach (8.3), aby otrzymać

$$c_i h_i/6 + c_{i+1}(h_i + h_{i+1})/3 + c_{i+1} h_{i+1}/6 \;=\; f(x_{i+1}, x_{i+2}) - f(x_i, x_{i+1}).$$

Wprowadzając oznaczenie

$$c_i^* \;=\; c_i/6, \tag{8.7}$$

możemy to równanie przepisać jako

$$\frac{h_i}{h_i + h_{i+1}} c_i^* \;+\; 2\,c_{i+1}^* \;+\; \frac{h_{i+1}}{h_i + h_{i+1}} c_{i+2}^* \;=\; f(x_i, x_{i+1}, x_{i+2}),$$

$0 \le i \le n-2$, albo w postaci macierzowej

$$\begin{pmatrix} 2 & w_1 & & & & \\ u_2 & 2 & w_2 & & & \\ & u_3 & 2 & w_3 & & \\ & & \ddots & \ddots & \ddots & \\ & & & u_{n-2} & 2 & w_{n-2} \\ & & & & u_{n-1} & 2 \end{pmatrix} \begin{pmatrix} c_1^* \\ c_2^* \\ c_3^* \\ \vdots \\ c_{n-2}^* \\ c_{n-1}^* \end{pmatrix} = \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ \vdots \\ v_{n-2} \\ v_{n-1} \end{pmatrix}, \qquad (8.8)$$

gdzie

$$\begin{aligned} u_i &= \frac{h_{i-1}}{h_{i-1}+h_i}, \qquad w_i = \frac{h_i}{h_{i-1}+h_i}, \\ v_i &= f(x_{i-1}, x_i, x_{i+1}). \end{aligned}$$

Ostatecznie, aby znaleźć współczynniki a_i, b_i, c_i, d_i należy najpierw rozwiązać układ równań (8.8), a potem zastosować wzory (8.7), (8.2), (8.6) i (8.5).

Zauważmy, że macierz układu (8.8) jest trójdiagonalna i ma dominującą przekątną. Układ można więc rozwiązać kosztem proporcjonalnym do wymiaru n używając algorytmu przeganiania z U. 3.6. Koszt znalezienia wszystkich współczynników kubicznej funkcji sklejanej interpolującej f jest więc też proporcjonalny do n.

Na końcu oszacujemy jeszcze błąd interpolacji naturalnymi kubicznymi funkcjami sklejanymi na przedziale $[a,b]$. Będziemy zakładać, że f jest dwa razy różniczkowalna w sposób ciągły.

Twierdzenie 8.3 *Jeśli $f \in F_M^1([a,b])$ to*

$$\|f - s_f\|_{\mathbf{C}([a,b])} \le 5\,M \max_{1\le i\le n} (x_i - x_{i-1})^2.$$

W szczególności, dla podziału równomiernego $x_i = a + (b-a)i/n$, $0 \le i \le n$, mamy

$$\|f - s_f\|_{\mathbf{C}([a,b])} \le 5\,M \left(\frac{b-a}{n}\right)^2.$$

Dowód Wykorzystamy obliczoną wcześniej postać interpolującej funkcji sklejanej s_f. Dla $x \in [x_i, x_{i+1}]$ mamy

$$\begin{aligned} w_i(x) &= f(x_i) + \left(\frac{f(x_{i+1}) - f(x_i)}{h_i} - h_i(c^*_{i+1} + 2c^*_i)\right)(x - x_i) \\ &\quad + 3c^*_i(x - x_i)^2 + \frac{c^*_{i+1} - c^*_i}{h_i}(x - x_i)^3. \end{aligned}$$

Z rozwinięcia f w szereg Taylora w punkcie x_i dostajemy $f(x) = f(x_i) + f'(x_i)(x - x_i) + f''(\xi_1)(x - x_i)^2/2$ oraz $(f(x_{i+1}) - f(x_i))/h_i = f'(x_i) + h_i f''(\xi_2)/2$. Stąd

$$\begin{aligned} f(x) - s_f(x) &= f(x) - w_i(x) \\ &= \frac{f''(\xi_1)}{2}(x - x_i)^2 - \left(\frac{f''(\xi_2)}{2} - (c^*_{i+1} + 2c^*_i)\right) h_i(x - x_i) \\ &\quad -3c^*_i(x - x_i)^2 - \frac{c^*_{i+1} - c^*_i}{h_i}(x - x_i)^3, \end{aligned}$$

oraz

$$\begin{aligned} |f(x) - s_f(x)| &\le (M + 2|c^*_{i+1}| + 6|c^*_i|)h_i^2 \\ &= (M + 8 \max_{1 \le i \le n-1} |c^*_i|)h_i^2. \end{aligned}$$

Niech teraz $\max_{1 \le i \le n-1} |c^*_i| = |c^*_s|$. Z postaci układu (8.8) otrzymujemy

$$\begin{aligned} |c^*_s| &= 2|c^*_s| - |c^*_s|(u_s + w_s) \le |u_s c^*_{s-1} + 2c^*_s + w_s c_{s+1}| \\ &= |f(x_{s-1}, x_s, x_{s+1})| \le \left|\frac{f''(\xi_3)}{2}\right| \le \frac{1}{2}M, \end{aligned}$$

a stąd i z (8.9)

$$|f(x) - s_f(x)| \le 5Mh_i^2,$$

co kończy dowód. □

Uwagi i uzupełnienia

U. 8.1 Zamiast terminu funkcje sklejane używa się też często terminów *splajny* (ang. *spline*-sklejać), albo *funkcje gięte*.

U. 8.2 W Rozdziale 7 pokazaliśmy, że aproksymacja kawałkami wielomianowa stopnia r jest optymalna (co do rzędu zbieżności) w klasie $F_M^r([a,b])$, wśród wszystkich aproksymacji korzystających jedynie z informacji o wartościach funkcji w danej liczbie n punktów. Okazuje się, że podobne optymalne własności wykazują funkcje sklejane, ale w innych klasach funkcji.

Niech

$$W_M^r(a,b) \,=\, \left\{ f \in W^r(a,b) : \int_a^b \left(f^{(r)}(x)\right)^2 dx \le M \right\}.$$

Ustalmy węzły $a = x_0 < \cdots < x_n = b$. Dla $f \in W_M^r(a,b)$, niech s_f będzie naturalną funkcją sklejaną interpolującą f w x_j, $0 \le j \le n$, a a_f dowolną inną aproksymacją korzystającą jedynie z informacji o wartościach f w tych węzłach , tzn.

$$a_f \,=\, \phi(f(x_0), \ldots, f(x_n)).$$

Załóżmy, że błąd aproksymacji mierzymy nie w normie Czebyszewa, ale w normie średniokwadratowej, zdefiniowanej jako

$$\|g\|_{\mathcal{L}_2(a,b)} \,=\, \sqrt{\int_a^b (g(x))^2\,dx}.$$

Wtedy

$$\sup_{f\in W_M^r(a,b)} \|f - s_f\|_{\mathcal{L}_2(a,b)} \,\le\, \sup_{f\in W_M^r(a,b)} \|f - a_f\|_{\mathcal{L}_2(a,b)}.$$

Aproksymacja naturalnymi funkcjami sklejanymi jest więc optymalna w klasie $W_M^r(a,b)$.

Można również pokazać, że interpolacja s_f^* naturalnymi funkcjami sklejanymi na węzłach równoodległych $x_j = a + (b-a)j/n$, $0 \le j \le n$, jest optymalna co do rzędu w klasie $W_M^r(a,b)$, wśród wszystkich aproksymacji korzystających jedynie z informacji o wartościach funkcji w $n+1$ dowolnych punktach, oraz

$$\max_{f\in W_M^r(a,b)} \|f - s_f^*\|_{\mathcal{L}_2(a,b)} \,\asymp\, n^{-r}.$$

U. 8.3 Tak jak wielomiany, naturalne funkcje sklejane interpolujące dane funkcje można reprezentować przez ich współczynniki w różnych bazach. Do tego celu można na przykład użyć bazy kanonicznej K_j, $0 \le j \le n$, zdefiniowanej równościami

$$K_j(x_i) \,=\, \begin{cases} 0 & i \ne j, \\ 1 & i = j, \end{cases}$$

przy której $s_f(x) = \sum_{j=0}^n f(x_j)K_j(x)$. Baza kanoniczna jest jednak niewygodna w użyciu, bo funkcje K_j w ogólności nie zerują się na żadnym podprzedziale, a tym samym manipulowanie nimi jest utrudnione.

Częściej używa się bazy B-sklejanej B_j, $0 \le j \le n$. W przypadku funkcji kubicznych, $r = 2$, jest ona zdefiniowana przez następujące warunki:

$$\begin{aligned} B_j(x_j) &= 1, && \text{dla } 0 \le j \le n, \\ B_j(x) &= 0, && \text{dla } x \le x_{j-2}, j \ge 2, \text{ oraz dla } x \ge x_{j+2}, j \le n-2. \end{aligned}$$

Dla B_0 i B_1 dodatkowo żądamy, aby

$$B_0''(x_0) \,=\, 0 \,=\, B_1''(x_0), \qquad B_1(x_0) \,=\, 0,$$

a dla B_{n-1} i B_n podobnie

$$B_{n-1}''(x_n) \,=\, 0 \,=\, B_n''(x_n), \qquad B_{n-1}(x_{n-1}) \,=\, 0.$$

Wtedy B_j nie zeruje się tylko na przedziale (x_{j-2}, x_{j+2}), Wyznaczenie współczynników rozwinięcia w bazie $\{B_i\}_{i=0}^n$ funkcji sklejanej interpolującej f wymaga rozwiązania układu liniowego z macierzą trójdiagonalną $\{B_j(x_i)\}_{i,j=0}^n$, a więc koszt obliczenia tych współczynników jest proporcjonalny do n.

U. 8.4 Oprócz naturalnych funkcji sklejanych często rozpatruje się też *okresowe funkcje sklejane*. Są to funkcje $\tilde{s} : \mathbf{R} \to \mathbf{R}$ spełniające warunki **(i)**, **(ii)** z Rozdziału 8.1, oraz warunek:

(iii)' $\tilde{s}^{(i)}$ jest dla $0 \le i \le r-1$ funkcją okresową o okresie $(b-a)$, tzn. $\tilde{s}^{(i)}(x) = \tilde{s}^{(i)}(x + (b-a))$, $\forall x$.

Klasę okresowych funkcji sklejanych rzędu r oznaczymy przez $\tilde{\mathcal{S}}_r$. Funkcje te mają podobne własności jak naturalne funkcje sklejane. Dokładniej, niech

$$\tilde{W}^r(a,b) \,=\, \{\, f \in W^r(a,b) :\, f^{(i)}(a) = f^{(i)}(b),\ 0 \le i \le r-1 \,\}, \tag{8.9}$$

tzn. $\tilde{W}^r(a,b)$ jest klasą funkcji z $W^r(a,b)$, które można przedłużyć do funkcji, krórych wszystkie pochodne do rzędu $r-1$ włącznie są $(b-a)$-okresowe na $\mathbf{R}$. Wtedy dla dowolnej funkcji $f \in \tilde{W}^r(a,b)$ zerującej się w węzłach x_j, oraz dla dowolnej $\tilde{s} \in \tilde{\mathcal{S}}_r$ mamy

$$\int_a^b f^{(r)}(x)\tilde{s}^{(r)}(x)\,dx \,=\, 0. \tag{8.10}$$

Jest to odpowiednik Lematu 8.1 w przypadku okresowym. W szczególności wynika z niego jednoznaczność rozwiązania zadania interpolacyjnego dla okresowych funkcji f (tzn. takich, że $f(a) = f(b)$), jak również odpowiednia własność minimalizacyjna okresowych funkcji sklejanych. Dokładniej, jeśli $f \in \tilde{W}^r(a,b)$ oraz $\tilde{s}_f \in \tilde{\mathcal{S}}_r$ interpoluje f w węzłach x_j, $0 \le j \le n$, to

$$\int_a^b \left(f^{(r)}(x)\right)^2 dx \;\ge\; \int_a^b \left(\tilde{s}_f^{(r)}(x)\right)^2 dx.$$

U. 8.5 Klasyczne zadanie aproksymacyjne w przestrzeniach funkcji definiuje się w następujący sposób.

Niech F będzie pewną przestrzenią liniową funkcji $f : [a,b] \to \mathbf{R}$, w której określona została norma $\|\cdot\|$. Niech $V_n \subset F$ będzie podprzestrzenią w F wymiaru n. Dla danej $f \in F$, należy znaleźć funkcję $v_f \in F$ taką, że

$$\|f - v_f\| \;=\; \min_{v \in V_n} \|f - v\|.$$

Okazuje się, że tak postawione zadanie ma rozwiązanie v_f, choć nie zawsze jest ono wyznaczone jednoznacznie, zob. Ćw. 8.6.

Jako przykład, rozpatrzmy $F = W^r(a,b)$. Utożsamiając funkcje $f_1, f_2 \in W^r(a,b)$ takie, że $f_1(x) - f_2(x) \in \Pi_{r-1}$, zdefiniujemy w $W^r(a,b)$ normę

$$\|f\| \;=\; \sqrt{\int_a^b \left(f^{(r)}(x)\right)^2 dx}.$$

Dla ustalonych węzłów $a = x_0 < \cdots < x_n = b$, niech

$$V_{n+1} \;=\; \mathcal{S}_r$$

będzie podprzestrzenią w $W^r(a,b)$ naturalnych funkcji sklejanych rzędu r opartych węzłach x_j, $0 \le j \le n$. Oczywiście $\dim\mathcal{S}_r = n+1$, co wynika z jednoznaczności rozwiązania w $\mathcal{S}_r$ zadania interpolacji. Okazuje się, że wtedy optymalną dla $f \in W^r(a,b)$ jest naturalna funkcja sklejana s_f interpolująca f w węzłach x_j, tzn.

$$\|f - s_f\| \;=\; \min_{s \in \mathcal{S}_r} \|f - s\|.$$

Rzeczywiście, ponieważ norma w przestrzeni $W^r(a,b)$ generowana jest przez iloczyn skalarny

$$\langle f_1, f_2 \rangle \;=\; \int_a^b f_1^{(r)}(x) f_2^{(r)}(x)\, dx,$$

jest to przestrzeń unitarna. Znane twierdzenie mówi, że w przestrzeni unitarnej najbliższą danej f funkcją w dowolnej domkniętej podprzestrzeni V jest rzut prostopadły f na V, albo równoważnie, taka funkcja $v_f \in V_{n+1}$, że iloczyn skalarny

$$\langle f - v_f, v\rangle \;=\; 0, \qquad \forall v \in V.$$

W naszym przypadku, ostatnia równość jest równoważna

$$\int_a^b (f - v_f)^{(r)}(x) s^{(r)}(x)\, dx \;=\; 0, \quad \forall s \in \mathcal{S}_r.$$

To zaś jest na mocy Lematu 8.1 prawdą gdy v_f interpoluje f w punktach x_j, czyli $v_f = s_f$.

Dodajmy jeszcze, że nie zawsze interpolacja daje najlepszą aproksymację w sensie klasycznym, zob. Ćw. 8.7.

Ćwiczenia

Ćw. 8.1 Dla $z \in \mathbf{R}$, niech

$$z_+ \;=\; \max(0, z) \;=\; \begin{cases} z & z \ge 0, \\ 0 & z < 0. \end{cases}$$

Pokazać, że każdą funkcję sklejaną s rzędu r można przedstawić w postaci

$$s(x) \;=\; w(x) \;+\; \sum_{j=0}^{n} a_j \Big((x - x_j)_+\Big)^{2r-1},$$

gdzie a_j są pewnymi współczynnikami rzeczywistymi, a $w \in \Pi_{2r-1}$. Ponadto, jeśli s jest naturalna, to $w \in \Pi_{r-1}$, a współczynniki a_j spełniają równania

$$\sum_{j=0}^{n} a_j x_j^i \;=\; 0, \qquad 0 \le i \le r - 1.$$

Ćw. 8.2 Niech $h > 0$ i $c \in \mathbf{R}$. Wyznaczyć współczynniki kubicznej funkcji sklejanej s opartej na pięciu węzłach $-2h, -h, 0, h, 2h$ i spełniającej dodatkowo następujące warunki interpolacyjne:

$$s(0) \;=\; c, \qquad s^{(k)}(\pm 2h) \;=\; 0, \qquad k = 0, 1, 2,$$

Ćw. 8.3 Wykorzystując Ćw. 8.2 wykazać, że dla dowolnego układu węzłów $a \le x_1 < \cdots x_n \le b$ mamy

$$\max\{\, \|f''\|_{\mathbf{C}([a,b])} \,:\, f \in F_M^1([a,b]), f(x_i) = 0, 1 \le i \le n\,\} \;\ge\; \frac{M}{24}\Big(\frac{b-a}{n+1}\Big)^2.$$

Stąd i z U. 7.6 wywnioskować, że dla każdej aproksymacji a_f korzystającej jedynie z wartości f w n punktach,

$$\max_{f \in F_M^1([a,b])} \|f - a_f\|_{\mathbf{C}([a,b])} \;\ge\; \frac{M}{24}\Big(\frac{b-a}{n+1}\Big)^2.$$

Ćw. 8.4 Pokazać, że funkcje B_j, $0 \le j \le n$, zdefiniowane w U. 8.3 istnieją i tworzą bazę w przestrzeni $\mathcal{S}_r$.

Ćw. 8.5 Pokazać jednoznaczność rozwiązania zadania interpolacyjnego w przypadku okresowych funkcji sklejanych (zob. U. 8.4), oraz własność minimalizacyjną (8.9).
Wskazówka. Zastosować technikę dowodową podobną do tej z przypadku naturalnych funkcji sklejanych.

Ćw. 8.6 Pokazać, że ogólnie postawione zadanie aproksymacyjne z U. 8.5 ma rozwiązanie.
Wskazówka. Zauważyć, że jeśli v_f istnieje to $\|v_f\| \le 2\|f\|$, oraz wykorzystać ciągłość odwzorowania $v \mapsto \|f - v\|$, $v \in V$, oraz fakt, że $\dim V < \infty$.

Ćw. 8.7 Niech $F = \mathbf{C}([a,b])$ z normą Czebyszewa, albo $F = \mathcal{L}_2(a,b)$ z normą średniokwadratową. Niech $V_{n+1} = \Pi_n$. Uzasadnić na przykładach, że w ogólności nie istnieją węzły x_j, $0 \le j \le n$, o następującej własności: dla każdej funkcji $f \in F$, wielomian $w_f \in \Pi_n$ interpolujący f w węzłach x_j jest elementem optymalnym w sensie aproksymacji klasycznej, zdefiniowanej w U. 8.5.

æ

Rozdział 9

Całkowanie numeryczne

Zajmiemy się teraz zadaniem całkowania numerycznego. Polega ono na obliczeniu (a raczej przybliżeniu) całki oznaczonej

$$S(f) = \int_a^b f(x)\,dx,$$

gdzie $-\infty < a < b < +\infty$, a f należy do pewnej klasy F funkcji rzeczywistych określonych i całkowalnych w sensie Riemanna na całym przedziale $[a, b]$.

Będziemy zakładać, że mamy możliwość obliczania wartości funkcji f, a w niektórych przypadkach również jej pochodnych, o ile istnieją. Dokładna całka $S(f)$ będzie więc w ogólności przybliżana wartością $A(f)$, która zależy tylko od wartości f i ew. jej pochodnych w skończonej liczbie punktów.

9.1 Co to są kwadratury?

Kwadraturami nazywamy funkcjonały liniowe $Q : F \to \mathbf{R}$ postaci

$$Q(f) = \sum_{i=0}^{n} a_i f(x_i),$$

albo ogólniej

$$Q(f) = \sum_{i=0}^{k} \sum_{j=0}^{n_i-1} a_{i,j} f^{(j)}(x_i), \tag{9.1}$$

gdzie x_i są punktami z $[a,b]$, a a_i (albo $a_{i,j}$) są pewnymi współczynnikami rzeczywistymi. Zauważmy, że obliczenia kwadratur są dopuszczalne w naszym modelu obliczeniowym, mogą więc służyć jako sposób przybliżania całki.

Jeden z możliwych sposobów konstrukcji kwadratur jest następujący. Najpierw wybieramy węzły x_j (pojedyncze lub wielokrotne), budujemy wielomian interpolacyjny odpowiadający tym węzłom, a następnie całkujemy go. Ponieważ postać wielomianu interpolacyjnego zależy tylko od danej informacji o f, otrzymana w ten sposób wartość też będzie zależeć tylko od tej informacji, a w konsekwencji funkcjonał wynikowy będzie postaci (9.1). Są to tzw. kwadratury interpolacyjne.

Definicja 9.1 *Kwadraturę Q^I opartą na węzłach o łącznej krotności $n+1$ nazywamy interpolacyjną, jeśli*

$$Q^I(f) \;=\; \int_a^b w_f(x)\,dx,$$

gdzie w_f jest wielomianem interpolacyjnym funkcji f stopnia co najwyżej n, opartym na tych węzłach.

Współczynniki kwadratur interpolacyjnych można łatwo wyliczyć. Rozpatrzmy dla uproszczenia przypadek, gdy węzły są jednokrotne. Zapisując wielomian interpolacyjny w postaci jego rozwinięcia w bazie kanonicznej Lagrange'a l_i (zob. (6.3)), otrzymujemy

$$Q^I(f) \;=\; \int_a^b \sum_{i=0}^n f(x_i) l_i(x)\,dx \;=\; \sum_{i=0}^n f(x_i) \int_a^b l_i(x)\,dx,$$

a stąd i z postaci l_i,

$$a_i \;=\; \int_a^b \frac{(x-x_0)\cdots(x-x_{i-1})(x-x_{i+1})\cdots(x-x_n)}{(x_i-x_0)\cdots(x_i-x_{i-1})(x_i-x_{i+1})\cdots(x_i-x_n)}\,dx,$$

$0 \le i \le n$.

Podamy teraz kilka przykładów.

Kwadratura prostokątów jest oparta na jednym węźle $x_0 = (a+b)/2$,

$$Q_0^I(f) \;=\; (b-a)f\Big(\frac{a+b}{2}\Big).$$

Kwadratura trapezów jest oparta na jednokrotnych węzłach $x_0 = a$, $x_1 = b$ i jest równa polu odpowiedniego trapezu,

$$Q_1^I(f) \;=\; T(f) \;=\; \frac{b-a}{2}\Big(f(a)+f(b)\Big).$$

Kwadratura parabol (Simpsona) jest oparta na jednokrotnych węzłach $x_0 = a$, $x_1 = b$, $x_2 = (a+b)/2$, i jest równa polu pod parabolą interpolującą f w tych węzłach,

$$Q_2^I(f) \;=\; P(f) \;=\; \frac{b-a}{6}\Big(f(a)+4f\Big(\frac{a+b}{2}\Big)+f(b)\Big).$$

Zauważmy, że kwadratury trapezów i parabol są oparte na węzłach jednokrotnych i równoodległych, przy czym $x_0 = a$ i $x_n = b$. Ogólnie, kwadratury interpolacyjne oparte na węzłach równoodległych $x_i = a + (b-a)i/n$, $0 \le i \le n$, nazywamy *kwadraturami Newtona–Cotesa.*

9.2 Błąd kwadratur interpolacyjnych

Zajmiemy się teraz błędem kwadratur interpolacyjnych. Przypomnijmy, że $F_M^n([a,b])$ oznacza klasę funkcji $(n+1)$ razy różniczkowalnych w sposób ciągły i takich, że $|f^{(n+1)}(x)| \le M$, $\forall x$.

Twierdzenie 9.1 *Niech Q^I będzie kwadraturą interpolacyjną opartą na (jednokrotnych lub wielokrotnych) węzłach x_i, $0 \le i \le n$. Jeśli $f \in F_M^n([a,b])$ to*

$$|S(f) \,-\, Q^I(f)| \;\le\; \frac{M}{(n+1)!}(b-a)^{n+2}. \tag{9.2}$$

W klasie $F_M^n([a,b])$, maksymalny błąd kwadratury Q^I wynosi

$$\sup_{f\in F_M^n([a,b])} |S(f)-Q^I(f)| \;=\; \frac{M}{(n+1)!}\int_a^b |(x-x_0)(x-x_1)\cdots(x-x_n)|\,dx.$$

Dowód Korzystając ze znanego nam już wzoru na błąd interpolacji wielomianowej z Lematu 7.1, mamy

$$S(f) \,-\, Q^I(f) \;=\; \int_a^b (x-x_0)(x-x_1)\cdots(x-x_n)f(x_0,x_1,\ldots,x_n,x)\,dx.$$

Stąd, jeśli $f \in F_M^n([a,b])$ to

$$|S(f) - Q^I(f)| \le \int_a^b (b-a)^{n+1} \frac{M}{(n+1)!}\, dx = (b-a)^{n+2} \frac{M}{(n+1)!}.$$

Ograniczenie górne w dokładnej formule na błąd w klasie $F_M^n([a,b])$ wynika bezpośrednio z oszacowania (9.2). Aby pokazać ograniczenie dolne zauważmy, że dla funkcji g takiej, że $g^{(n+1)}$ przyjmuje na przedziałach (a, x_0), (x_0, x_1), ..., (x_n, b) naprzemiennie wartości M i $-M$ mamy

$$|S(g) - Q^I(g)| = \frac{M}{(n+1)!} \int_a^b |(x-x_0)(x-x_1)\cdots(x-x_n)|\, dx.$$

Co prawda, g nie jest w $F_M^n[a,b]$, ale może być dla dowolnego $\varepsilon > 0$ przybliżana funkcjami $f_\varepsilon \in F_M^n([a,b])$ w ten sposób, że całka

$$\int_a^b |(x-x_0)\cdots(x-x_n)(f-g)^{(n+1)}(x)|\, dx \le \varepsilon\,(n+1)!.$$

Zapisując $f_\varepsilon = g + (f_\varepsilon - g)$ mamy

$$\begin{aligned} |S(f_\varepsilon) - Q^I(f_\varepsilon)| &\le |S(g) - Q^I(g)| + |S(f_\varepsilon - g) - Q^I(f_\varepsilon - g)| \\ &\le \frac{M}{(n+1)!} \int_a^b |(x-x_0)\cdots(x-x_n)|\, dx + \varepsilon, \end{aligned}$$

co wobec dowolności ε daje dowód twierdzenia. □

W szczególnych przypadkach kwadratur trapezów T i parabol P możemy otrzymać innego rodzaju formuły na błąd.

Twierdzenie 9.2 *(i) Jeśli $f \in \mathbf{C}^{(2)}([a,b])$ to dla kwadratury trapezów mamy*

$$S(f) - T(f) = -\frac{(b-a)^3}{12} f^{(2)}(\xi_1).$$

(ii) Jeśli $f \in \mathbf{C}^{(4)}([a,b])$ to dla kwadratury parabol mamy

$$S(f) - P(f) = -\frac{(b-a)^5}{2280} f^{(4)}(\xi_2).$$

($\xi_1, \xi_2 \in [a,b]$).

Dowód (i) Ze wzoru (9.2)

$$S(f) - T(f) = \int_a^b (x-a)(x-b)f(a,b,x)\,dx.$$

Ponieważ funkcja $x \mapsto f(a,b,x)$ jest ciągła, a wielomian $(x-a)(x-b)$ przyjmuje jedynie wartości nieujemne, można zastosować twierdzenie o wartości średniej dla całki, aby otrzymać

$$\begin{aligned} S(f) - T(f) &= f(a,b,c)\int_a^b (x-a)(x-b)\,dx \\ &= -\frac{f^{(2)}(\xi_1)}{2!}\frac{(b-a)^3}{6}, \end{aligned}$$

dla pewnych $c, \xi_1 \in [a,b]$.

(ii) Niech $w_{f,2} \in \Pi_2$ i $w_{f,3} \in \Pi_3$ będą wielomianami interpolacyjnymi funkcji f odpowiednio dla węzłów $a, b, (a+b)/2$ oraz $a, b, (a+b)/2, (a+b)/2$. Wtedy

$$w_{f,3}(x) = w_{f,2}(x) + f\Big(a,b,\frac{a+b}{2},\frac{a+b}{2}\Big)(x-a)\Big(x-\frac{a+b}{2}\Big)(x-b).$$

Wobec

$$\int_a^b (x-a)\Big(\frac{a+b}{2}\Big)(x-b)\,dx = 0$$

mamy

$$P(f) = \int_a^b w_{f,2}(x)\,dx = \int_a^b w_{f,3}(x)\,dx.$$

Stąd i ze wzoru na błąd interpolacji Hermite'a otrzymujemy

$$\begin{aligned} S(f) - P(f) &= \int_a^b (f - w_{f,3})(x)\,dx \\ &= \int_a^b (x-a)\Big(x-\frac{a+b}{2}\Big)^2(x-b)f\Big(a,b,\frac{a+b}{2},\frac{a+b}{2},x\Big)\,dx. \end{aligned}$$

Ponieważ wielomian $(x-a)(x-(a+b)/2)^2(x-b)$ jest niedodatni na $[a,b]$, możemy znów zastosować twierdzenie o wartości średniej. Mamy

$$\begin{aligned} S(f) - P(f) &= f\Big(a,b,\frac{a+b}{2},\frac{a+b}{2},c\Big) \\ &\qquad \int_a^b (x-a)\Big(x-\frac{a+b}{2}\Big)^2(x-b)\,dx \\ &= -\frac{f^{(4)}(\xi_2)}{4!}\frac{(b-a)^5}{120}, \end{aligned}$$

co kończy dowód. □

9.3 Kwadratury złożone

Podobnie jak w przypadku zadania interpolacji chcielibyśmy, aby błąd kwadratur malał do zera, gdy liczba węzłów rośnie do nieskończoności. Można to osiągnąć stosując np. *kwadratury złożone*. Są to kwadratury, które powstają przez scałkowanie funkcji kawałkami wielomianowej interpolującej f.

Prostym przykładem kwadratury złożonej jest suma Riemanna,

$$\bar{Q}(f) = \sum_{i=0}^{n}(t_{i+1} - t_i)f(x_i),$$

gdzie $a = t_0 < t_1 < \cdots < t_{n+1} = b$ oraz $x_i \in [t_i, t_{i+1}]$. Jeśli średnica podziału, $\max_{0\le i\le n}(t_i - t_{i-1})$, maleje do zera to $\lim_{n\to\infty} \bar{Q}(f) = S(f)$.

Będziemy rozpatrywać kwadratury złożone postaci

$$\bar{Q}(f) = \int_a^b \bar{w}_f(x)\,dx,$$

gdzie $\bar{w}_f$ jest kawałkami wielomianem z Rozdziału 7.3. To znaczy, dla danego n kładziemy $t_i = a + (b-a)i/k$, $0 \le i \le k$, a następnie dla każdego i wybieramy dowolne węzły $x_{i,j} \in [t_{i-1}, t_i]$, $0 \le j \le r$. Wtedy $\bar{w}_f$ jest na każdym przedziale wielomianem interpolacyjnym funkcji f stopnia co najwyżej r opartym na węzłach $x_{i,j}$. Kwadratura $\bar{Q}$ korzysta z węzłów o łącznej krotności $n \le k(r+1)$.

Twierdzenie 9.3 *Błąd kwadratury złożonej $\bar{Q}(f)$ w klasie $F_M^r([a,b])$ jest ograniczony przez*

$$\sup_{f\in F_M^r([a,b])} |S(f) - \bar{Q}(f)| \le \frac{(b-a)^{r+2}}{k^{r+1}}\frac{M}{(r+1)!} \le C\left(\frac{1}{n}\right)^{r+1},$$

gdzie

$$C = \frac{M(r+1)^{r+1}(b-a)^{r+2}}{(r+1)!}.$$

Dowód Twierdzenie to jest bezpośrednim wnioskiem z Twierdzenia 9.1. Mamy bowiem

$$\begin{aligned} |S(f) - \bar{Q}(f)| &\leq \sum_{i=1}^{k} \int_{t_{i-1}}^{t_i} |f(x) - \bar{w}_f(x)|\, dx \\ &\leq \sum_{i=1}^{k} \Big(\frac{b-a}{k}\Big)^{r+2} \frac{M}{(r+1)!} = \frac{(b-a)^{r+2}}{k^{r+1}} \frac{M}{(r+1)!}, \end{aligned}$$

co kończy dowód. □

W klasie $F_M^r([a,b])$, błąd kwadratur złożonych jest rzędu $n^{-(r+1)}$, czyli jest taki sam jak błąd interpolacji kawałkami wielomianowej. I tak jak przy interpolacji można pokazać, że błąd każdej innej metody całkowania korzystającej jedynie z wartości funkcji w n punktach nie może w klasie $F_M^r([a,b])$ maleć szybciej niż $n^{-(r+1)}$, zob. U. 9.1. Podane kwadratury złożone mają więc optymalny rząd zbieżności.

Zajmiemy się teraz błędem szczególnych kwadratur złożonych, mianowicie złożonych kwadratur trapezów $\bar{T}_k$ i parabol $\bar{P}_k$. Powstają one przez zastosowanie na każdym przedziale $[t_{i-1}, t_i]$ odpowiednio kwadratur trapezów T i parabol P. Jak łatwo się przekonać,

$$\bar{T}_k(f) = \frac{b-a}{k} \left(\frac{f(a)+f(b)}{2} + \sum_{j=1}^{k-1} f\Big(\frac{j}{k}\Big) \right),$$

oraz

$$\bar{P}_k(f) = \frac{b-a}{3k} \left(\frac{f(a)+f(b)}{2} + \sum_{j=1}^{k-1} f\Big(\frac{j}{k}\Big) + 2 \sum_{j=1}^{k} f\Big(\frac{2j-1}{2k}\Big) \right).$$

Twierdzenie 9.4 *(i) Jeśli $f \in \mathbf{C}^{(2)}([a,b])$ to*

$$S(f) - \bar{T}_k(f) = -\frac{(b-a)^2}{12\, k^2} f^{(2)}(\xi_1).$$

(ii) Jeśli $f \in \mathbf{C}^{(4)}([a,b])$ to

$$S(f) - \bar{P}_k(f) = -\frac{(b-a)^4}{2280\, k^4} f^{(4)}(\xi_2).$$

Dowód Dla kwadratury trapezów mamy

$$\begin{aligned} S(f) - \bar{T}_k(f) &= -\sum_{i=1}^{k} \frac{(b-a)^3}{12k^3} f^{(2)}(\alpha_i) \\ &= -\frac{(b-a)^2}{12k^2} \sum_{i=1}^{k} \frac{b-a}{k} f^{(2)}(\alpha_i) = -\frac{(b-a)^2}{12k^2} f^{(2)}(\xi_1), \end{aligned}$$

a dla kwadratury parabol podobnie

$$\begin{aligned} S(f) - \bar{P}_k(f) &= -\sum_{i=1}^{k} \frac{(b-a)^5}{2280k^5} f^{(4)}(\beta_i) \\ &= -\frac{(b-a)^4}{2280k^4} \sum_{i=1}^{k} \frac{b-a}{k} f^{(4)}(\beta_i) = -\frac{(b-a)^4}{2280k^4} f^{(4)}(\xi_2). \quad \square \end{aligned}$$

Kwadratura parabol ma więc optymalny rząd zbieżności nie tylko w klasie $F_M^2([a,b])$, ale też w $F_M^3([a,b])$.

9.4 Przyspieszanie zbieżności kwadratur

W praktyce często stosuje się obliczanie kwadratur poprzez zagęszczanie podziału przedziału $[a,b]$. Na przykład, dla złożonej kwadratury trapezów zachodzi następujący wygodny wzór rekurencyjny:

$$\bar{T}_{2k} = \frac{1}{2}\left(\bar{T}_k(f) + \frac{b-a}{k} \sum_{i=1}^{k} f\Big(\frac{2i-1}{2k}\Big)\right). \tag{9.3}$$

Pozwala on obliczyć $\bar{T}_{2k}(f)$ na podstawie $\bar{T}_k(f)$ poprzez "doliczenie" wartości funkcji w punktach "gęstszej" siatki. W ten sposób możemy obserwować zachowanie się kolejnych przybliżeń $\bar{T}_{2^s}(f)$ $(s \geq 0)$ całki $S(f)$. Jest to szczególnie istotne wtedy, gdy nie mamy żadnej informacji a priori o $\|f''\|_{\mathbf{C}([a,b])}$, a przez to nie potrafimy oszacować liczby n węzłów, dla której osiągniemy pożądaną dokładność, zob. U. 9.2.

Jeśli funkcja jest więcej niż dwa razy różniczkowalna to użycie złożonych kwadratur trapezów zdaje się tracić sens. Wtedy istnieją przecież kwadratury, których błąd maleje do zera szybciej niż n^{-2}. Okazuje się

jednak, że kwadratury $\bar{T}_k$ mogą być podstawą dla prostej rekurencyjnej konstrukcji innych kwadratur posiadających już optymalną zbieżność. Konstrukcja ta bazuje na następującym ważnym lemacie.

Lemat 9.1 *(Formuła Eulera-Maclaurina)*
Dla funkcji $f \in \mathbf{C}^{(2m+2)}([a,b])$, błąd złożonej kwadratury trapezów $\bar{T}_k$ wyraża się wzorem

$$\begin{aligned} S(f) - \bar{T}_k(f) &= \sum_{i=1}^{m} c_i h^{2i}\Big(f^{(2i-1)}(b) - f^{(2i-1)}(a)\Big) \\ &\qquad + c_{m+1} h^{2m+2}(b-a) f^{(2m+2)}(\xi_{m,k}), \end{aligned}$$

gdzie $h = (b-a)/k$, $\xi_{m,k} \in [a,b]$, a c_i są pewnymi stałymi liczbowymi. Mamy $c_1 = -1/12$, $c_2 = -1/720$ i, ogólnie, $c_i = B_i/(2i)!$, gdzie B_i są tzw. liczbami Bernoulliego. $\square$

Dowód tego lematu pominiemy.

Formułę Eulera-Maclaurina można przepisać w postaci

$$S(f) - \bar{T}_k(f) = \sum_{i=1}^{m} c_i^{(0)}(f)\, k^{-2i} + c_{m+1,k}^{(0)}(f)\, k^{-(2m+2)},$$

gdzie $c_i^{(0)}(f) = c_i(b-a)^{2i}(f^{(2i-1)}(b) - f^{(2i-1)}(a))$, $1 \le i \le m$, oraz $c_{m+1,k}^{(0)}(f) = c_{m+1}(b-a)^{2m+2} f^{(2m+2)}(\xi_{m+1,k})$. Zauważmy przy tym, że jeśli $f \in F_M^{2m+1}([a,b])$ to współczynniki $c_{m+1,k}^{(0)}(f)$ są wspólnie ograniczone przez $c_{m+1}(b-a)^{2m+2}M$.

Definiując teraz kwadraturę

$$\bar{T}_k^1(f) = \frac{4\,\bar{T}_{2k}(f) - \bar{T}_k(f)}{3},$$

dla $f \in \mathbf{C}^{(4)}([a,b])$ mamy

$$\begin{aligned} S(f) - \bar{T}_k^1(f) &= \frac{4\,(S(f) - \bar{T}_{2k}(f) - (S(f) - \bar{T}_k(f))}{3} \\ &= \frac{4}{3}\left(\frac{c_1^{(0)}(f)}{4k^2} + \frac{c_{2,2k}^{(0)}(f)}{4^2k^4}\right) - \frac{1}{3}\left(\frac{c_1^{(0)}(f)}{k^2} + \frac{c_{2,k}^{(0)}(f)}{k^4}\right) \\ &= \frac{c_{2,k}^{(1)}(f)}{k^4}, \end{aligned}$$

gdzie $c_{2,k}^{(1)}(f) = (1/12)c_{2,2k}^{(0)}(f) - (1/3)c_{2,k}^{(0)}(f)$ i jest wspólnie ograniczone dla $f \in F_M^3([a,b])$. Kwadratura T_k^1 ma więc optymalny w $F_M^3([a,b])$ rząd zbieżności k^{-4}. Proces ten można kontynuować dalej tworząc kolejne kwadratury o coraz to wyższym rzędzie zbieżności. Dokładniej, połóżmy $\bar{T}_k^0(f) = \bar{T}_k(f)$ oraz, dla $s \geq 1$,

$$\bar{T}_k^s(f) \;=\; \frac{4^s\,\bar{T}_{2k}^{s-1}(f) \;-\; \bar{T}_k^{s-1}(f)}{4^s - 1}. \tag{9.4}$$

Wtedy, dla $f \in F_M^{2m+1}([a,b])$, rząd zbieżności kwadratury $\bar{T}_k^m$ wynosi $k^{-(2m+2)}$. Rzeczywiście, sprawdziliśmy, że jest to prawdą dla $m = 0, 1$. Niech $m \geq 2$. Postępując indukcyjnie ze względu na $s = 1, 2, \ldots, m$ mamy

$$\begin{aligned} S(f) \;-\; \bar{T}_k^s(f) \;&=\; \frac{4^s(S(f) - \bar{T}_{2k}^{s-1}(f)) - (S(f) - \bar{T}_k^{s-1}(f))}{4^s \;-\; 1} \\ &=\; \left(4^s\left(\sum_{i=s}^{m} c_i^{(s-1)}(f)(2k)^{-2i} + c_{m+1,2k}^{(s-1)}(f)(2k)^{-(2m+2)}\right)\right. \\ &\qquad \left. -\left(\sum_{i=s}^{m} c_i^{(s-1)}(f)k^{-2i} + c_{m+1,k}^{(s-1)}(f)k^{-(2m+2)}\right)\right)\frac{1}{4^s \;-\; 1} \\ &=\; \sum_{i=s+1}^{m} c_i^{(s)}(f)k^{-2i} \;+\; c_{m+1,k}^{(s)}(f)k^{-(2m+2)}, \end{aligned}$$

ponieważ współczynniki przy k^{-2s} redukują się. $c_i^{(s)}(f)$ są tutaj pewnymi nowymi stałymi, a $c_{m+1,k}^{(s)}(f)$ może być w klasie $F_M^{2m+1}([a,b])$ ograniczona przez stałą niezależną od f. Ostatecznie, dla $s = m$ mamy więc

$$S(f) \;-\; \bar{T}_k^m(f) \;=\; c_{m+1,k}^{(m)}(f)k^{-(2m+2)}$$

i w klasie $F_M^{2m+1}([a,b])$

$$|S(f) \;-\; \bar{T}_k^m(f)| \;\leq\; c_m\, k^{-(2m+2)}$$

dla pewnej stałej c_m niezależnej od f.

Zauważmy jeszcze, że $\bar{T}_k^m$ wykorzystuje $n = k2^m + 1$ wartości f w punktach równoodległych na $[a,b]$ co oznacza, że w terminach n rząd zbieżności wynosi też $n^{-(2m+2)}$, a więc jest optymalny w klasie $F_M^{2m+1}([a,b])$.

Kwadratury $\bar{T}_k^s$ nazywane są *kwadraturami Romberga*. Dla danej funkcji f, można je łatwo konstruować budując następującą tablicę trójkątną:

$$\begin{array}{llllll}
\bar{T}_1^0(f) \\
\bar{T}_2^0(f) & \bar{T}_1^1(f) \\
\bar{T}_4^0(f) & \bar{T}_2^1(f) & \bar{T}_1^2(f) \\
\bar{T}_8^0(f) & \bar{T}_4^1(f) & \bar{T}_2^2(f) & \bar{T}_1^3(f) \\
\vdots & \vdots & \vdots & \vdots & \ddots \\
\bar{T}_{2^s}^0(f) & \bar{T}_{2^{s-1}}^1(f) & \bar{T}_{2^{s-2}}^2(f) & \bar{T}_{2^{s-3}}^3(f) & \cdots & \bar{T}_1^s(f),
\end{array} \tag{9.5}$$

w której pierwsza kolumna jest tworzona indukcyjnie zgodnie ze wzorem (9.3), a kolejne zgodnie z (9.4).

Uwagi i uzupełnienia

U. 9.1 Pokażemy, że jeśli dowolny algorytm A daje przybliżoną wartość całki $S(f) = \int_a^b f(x)dx$ na podstawie jedynie wartości lub pochodnych funkcji f w n różnych punktach, tzn.

$$A(f) \;=\; \phi(f(x_1), \ldots, f^{(r+1)}(x_1), \ldots, f(x_n), \ldots, f^{(r+1)}(x_n))$$

dla pewnych $x_i \in [a,b]$, to błąd takiego przybliżenia w klasie $F_M^r([a,b])$,

$$\sup_{f \in F_M^r([a,b])} |S(f) \,-\, A(f)| \;\ge\; C_1\, M\, n^{-(r+1)}, \tag{9.6}$$

gdzie C_1 jest pewną stałą dodatnią niezależną od A i n.

W tym celu, podobnie jak w U. 7.2, wybierzmy dowolną nieujemną funkcję $\psi : \mathbf{R} \to \mathbf{R}$ spełniającą warunki:

1. $\psi \in F_1^r(\mathbf{R})$,

2. $\psi(x) = 0$ dla $x \notin [0,1]$.

Oznaczmy $x_0 = a$, $x_{n+1} = b$, $h_i = x_{i+1} - x_i$, $0 \le i \le n$, oraz zdefiniujmy funkcje

$$\psi_i(x) \;=\; M h_i^{r+1} \psi\Big(\frac{x - x_i}{h_i}\Big).$$

Wtedy dla $g_1(x) = -g_2(x) = \sum_{i=0}^n \psi_i(x)$ mamy $g_l \in F_M^r([a,b])$ oraz informacja o g_l jest zerowa, $g_l^{(p)}(x_j) = 0$ dla $0 \le p \le r+1$, $1 \le j \le n$. Stąd $\phi_0 = \phi(\underbrace{0,\ldots,0}_{n(r+2)})$ jest aproksymacją dla całek obu funkcji ψ_l i

$$\begin{aligned}\max\{|S(g_1)-\phi_0|, |S(g_2)-\phi_0|\} \;&\ge\; |S(g_1)-S(g_2)|/2 \;=\; S(g_1) \\ =\; \sum_{i=0}^n \int_{x_{i-1}}^{x_i} \psi_i(x)\,dx \;&=\; M\sum_{i=0}^n h_i^{r+1} \int_{x_{i-1}}^{x_i} \psi\Big(\frac{x-x_i}{h_i}\Big)\,dx \\ =\; M\int_0^1 \psi(x)\,dx \;+\; \sum_{i=0}^n h_i^{r+2}.\end{aligned}$$

Zauważmy jeszcze, że suma $\sum_{i=0}^n h_i^{r+2}$ jest minimalna dla $h_i = (b-a)/(n+1)$, $\forall i$, a stąd

$$S(g_1) \;\ge\; M\frac{(b-a)^{r+2}}{(n+1)^{r+1}} \int_0^1 \psi(x)\,dx.$$

Nierówność (9.6) zachodzi więc z $C_1 = (2(b-a))^{r+1}\int_0^1 \psi(x)dx$.

U. 9.2 Jeśli $f \in \mathbf{C}^{(2)}([a,b])$ i $f'(a) \ne f'(b)$ to dla dużych k mamy

$$S(f) \;-\; \bar{T}_k(f) \;\approx\; c(f)k^{-2},$$

gdzie $c(f) = -(b-a)^2(f'(b)-f'(a))/12$. Stąd

$$\begin{aligned}\bar{T}_k(f) \;-\; \bar{T}_{k/2}(f) \;&=\; (S(f)-\bar{T}_{k/2}(f)) \;-\; (S(f)-\bar{T}_k(f)) \\ &\approx\; \frac{c(f)}{k^2} \;-\; \frac{c(f)}{(k/2)^2} \;=\; -\frac{3c(f)}{k^2} \;\approx\; -3\,(S(f)-\bar{T}_k(f)),\end{aligned}$$

a więc

$$S(f) \;-\; \bar{T}_k(f) \;\approx\; \frac{1}{3}\,(\bar{T}_{k/2}(f)-\bar{T}_k(f)),$$

gdy $k \to \infty$. Ostatnia przybliżona równość może sugerować kryterium kończenia obliczeń, gdy chcemy otrzymać przybliżenie całki $S(f)$ z zadaną dokładnością $\varepsilon > 0$. Mianowicie, obliczając kolejne kwadratury $\bar{T}_{2^s}(f)$ dla $s = 0, 1, \ldots$, sprawdzamy jednocześnie, czy

$$|\bar{T}_{2^{s-1}}(f) - \bar{T}_{2^s}(f)| \;\le\; 3\,\varepsilon.$$

Jeśli ten warunek jest spełniony dla kilku kolejnych s to z dużym prawdopodobieństwem możemy stwierdzić, że $|S(f) - \bar{T}_{2^s}(f)| \le \varepsilon$.

Oczywiście, podobne kryteria kończenia obliczeń mogą być konstruowane dla innych ciągów kwadratur, jeśli tylko znamy teoretyczne zachowanie się błędu.

Ćwiczenia

Ćw. 9.1 Pokazać, że w kwadraturze Newtona-Cotesa opartej na $n+1$ równoodległych węzłach $x_i = a + (b-a)i/n$, współczynniki a_i dane są wzorami

$$a_i \;=\; \frac{(-1)^{n-i}(b-a)}{n\,i!(n-i)!}\int_0^n x(x-1)\cdots(x-(i-1))(x-(i+1))\cdots(x-n)\,dx.$$

Ćw. 9.2 Pokazać, że jeśli $f \in \mathbf{C}^{(2)}([a,b])$ to dla kwadratury prostokątów $Q_0(f) = f((a+b)/2)(b-a)/2$ mamy

$$S(f) \;-\; Q_0(f) \;=\; \frac{(b-a)^3}{24} f^{(2)}(\xi_0),$$

($\xi_0 \in [a,b]$), a w konsekwencji

$$\max_{f\in F_M^1([a,b])} |S(f) - Q_0(f)| \;=\; \frac{M(b-a)^3}{24}.$$

Błąd w klasie $F_M^1([a,b])$ jest więc dla kwadratury prostokątów dwa razy mniejszy niż dla kwadratury trapezów. Wywnioskować stąd, że złożona kwadratura prostokątów,

$$\bar{Q}_0(f) \;=\; \frac{b-a}{k}\sum_{i=1}^{k} f\Big(\frac{2i-1}{2k}\Big),$$

ma optymalny rząd zbieżności nie tylko w klasie $F_M^0([a,b])$, ale też w klasie $F_M^1([a,b])$.

Ćw. 9.3 Rozpatrzmy kwadratury interpolacyjne oparte na dwóch węzłach $x_0, x_1 \in [a,b]$. Pokazać, że wśród tych kwadratur najmniejszy błąd w klasie $F_M^1([a,b])$ jest osiągany przez kwadraturę

$$Q^I(f) \;=\; \frac{b-a}{2}\left(f\Big(\frac{3a+b}{4}\Big) + f\Big(\frac{a+3b}{4}\Big)\right),$$

a jej błąd

$$\sup_{f\in F_M^1([a,b])} |S(f) - Q^I(f)| \;=\; \frac{M(b-a)^3}{32}.$$

Ćw. 9.4 Uzasadnić, że jeśli kwadratura interpolacyjna Q^I oparta jest na węzłach Czebyszewa

$$x_i \,=\, \frac{a+b}{2}+\frac{b-a}{2}\cos\Big(\frac{2i+1}{2r+2}\pi\Big), \quad 0\le i\le r,$$

to dla $f\in F_M^r([a,b])$ błąd

$$|S(f)\,-\,Q^I(f)| \,\le\, \frac{M(b-a)^{r+2}}{2^{2r+1}(r+1)!},$$

a dla odpowiadającej węzłom Czebyszewa kwadratury złożonej

$$|S(f)\,-\,\bar{Q}^I(f)| \,\le\, \frac{M(b-a)^{r+2}}{2^{2r+1}(r+1)!}\Big(\frac{1}{k}\Big)^{r+1}.$$

Ćw. 9.5 Rozpatrzmy kwadraturę

$$\bar{Q}_k(f) \,=\, \bar{T}_k(f) \,-\, \frac{(b-a)^2}{12k^2}(f'(b)-f'(a)).$$

Pokazać, nie korzytając z formuły Eulera-Maclaurina, że jeśli $f\in F_M^2([a,b])$ to błąd $|S(f)-\bar{Q}_k(f)|$ zbiega do zera co najmniej tak szybko jak k^{-4}, gdy $k\to\infty$. W szczególności, dla funkcji spełniających $f'(a)=f'(b)$ rząd zbieżności złożonej kwadratury trapezów wynosi k^{-4}.

Ćw. 9.6 Opracować ekonomiczny algorytm obliczania ciągu kolejnych złożonych kwadratur parabol $\bar{P}_{2^s}(f)$ dla $s=0,1,2,\ldots$.

Ćw. 9.7 Opracować kryterium kończenia obliczeń analogiczne do tego z U. 9.2, dla ciągu złożonych kwadratur parabol $\bar{P}_{2^s}(f)$, $s=0,1,\ldots$.

Ćw. 9.8 Pokazać, że drugą kolumnę tabeli (9.5) kwadratur Romberga tworzą złożone kwadratury parabol, tzn.

$$\bar{P}_k(f) \,=\, \frac{4\bar{T}_{2k}(f)-\bar{T}_k(f)}{3}.$$

Ćw. 9.9 Opracować program obliczający wartość $\bar{T}_1^s(f)$ kwadratury Romberga, korzystając ze wzorów (9.3) i budując tabelę (9.4).

æ

Rozdział 10

Całkowanie a aproksymacja

W poprzednim rozdziale badaliśmy kwadratury ze względu na kryterium błędu w pewnych klasach funkcji. W tym rozdziale zajmiemy się badaniem kwadratur ze względu na inne, bardziej klasyczne kryterium, którym jest rząd kwadratury. Będziemy przy tym rozpatrywać nieco ogólniejsze zadanie całkowania, tzw. całkowanie z wagą,

$$S(f) \,=\, S_\rho(f) \,=\, \int_a^b f(x)\rho(x)\,dx,$$

gdzie *waga* ρ jest prawie wszędzie dodatnia i jest całkowalna,

$$\int_a^b |\rho(x)|\,dx \,<\, +\infty.$$

Dopuścimy również możliwość nieskończonych przedziałów całkowania, a więc $-\infty \le a < b \le +\infty$.

10.1 Rząd kwadratury

Zaczniemy od definicji.

Definicja 10.1 *Rzędem kwadratury Q nazywamy taką liczbę r, że*

(i) *kwadratura Q daje dokładną wartość całki dla wszystkich wielomianów stopnia mniejszego niż r,*

$$Q(w) \,=\, S_\rho(w), \qquad \forall w \in \Pi_{r-1},$$

(i) *istnieje wielomian stopnia r dla którego kwadratura nie jest równa całce,*

$$\exists w^* \in \Pi_r, \qquad Q(w^*) \neq S_\rho(w^*).$$

Rząd kwadratury Q oznaczymy przez $\mathrm{rz}(Q)$. Dla przykładu, załóżmy, że $\rho \equiv 1$ i przedział $[a, b]$ jest skończony. Wtedy rząd kwadratury prostokątów wynosi 2, tak jak rząd kwadratury trapezów T, natomiast rząd kwadratury parabol P wynosi 4. Rzeczywiście, to wynika bezpośrednio ze wzorów na błędy tych kwadratur danych w Ćw. 9.2 i Twierdzeniu 9.2. Ogólniej, mamy następujące twierdzenie, które przytaczamy bez dowodu.

Twierdzenie 10.1 *Niech waga $\rho \equiv 1$ i niech przedział $[a, b]$ będzie skończony. Rząd kwadratury Newtona-Cotesa Q_n^{NC}, tzn. opartej na jednokrotnych węzłach równoodległych $x_i = a + (b - a)i/n$, $0 \le i \le n$, wynosi*

$$\mathrm{rz}(Q^{NC}) = \begin{cases} n+1 & \textit{dla } n \textit{ nieparzystych,} \\ n+2 & \textit{dla } n \textit{ parzystych.} \end{cases}$$

□

Jasne jest, że interesują nas kwadratury o jak najwyższym rzędzie. Chcielibyśmy wiedzieć, jaki jest maksymalny rząd kwadratury korzystającej z ustalonej liczby węzłów i jak konstruować kwadratury o maksymalnym rzędzie. Jak przekonamy się później, nie jest to tylko akademickie pytanie, bowiem kwadratury o maksymalnym rzędzie mają również dobre własności ze względu na błąd.

Częściową odpowiedź na pytanie o maksymalny rząd kwadratur daje następujący lemat.

Lemat 10.1 *Niech Q będzie kwadraturą opartą na węzłach o łącznej krotności $n + 1$.*
(i) Jeśli $\mathrm{rz}(Q) \ge n + 1$ *to Q jest kwadraturą interpolacyjną.*
(ii) Jeśli Q jest kwadraturą interpolacyjną to

$$n + 1 \le \mathrm{rz}(Q) \le 2(n + 1).$$

Dowód (i) Dla dowolnej funkcji f, niech w_f będzie wielomianem interpolacyjnym dla f stopnia co najwyżej n, opartym na tej samej informacji o f, z jakiej korzysta kwadratura Q. Wtedy mamy $Q(f) = Q(w_f)$, a ponieważ $\mathrm{rz}(Q) \geq n+1$ to również $Q(w_f) = S_\rho(w_f)$. Stąd

$$Q(f) \;=\; S_\rho(w_f),$$

a to oznacza, że Q jest interpolacyjna.

(ii) Jeśli Q jest interpolacyjna to dla każdego $w \in \Pi_n$ mamy $Q(w) = S_\rho(w)$, bo w jest dla siebie wielomianem interpolacyjnym. Stąd $\mathrm{rz}(Q) \geq n+1$.

Dla dowodu drugiej nierówności załóżmy, że Q korzysta z węzłów $x_0, x_1, \ldots, x_n$ (być może powtarzających się). Wtedy dla wielomianu

$$w_1(x) \;=\; (x-x_0)^2(x-x_1)^2\cdots(x-x_n)^2$$

mamy $w_1 \in \Pi_{2(n+1)}$ oraz $S_\rho(w_1) > 0$, bo w_1 jest prawie wszędzie dodatni. Z drugiej strony, $Q(w_1) = 0$, bo informacja o w_1 jest zerowa. Stąd

$$Q(w_1) \;\neq\; S_\rho(w_1)$$

i w konsekwencji $\mathrm{rz}(Q) \leq 2(n+1)$. □

Lemat 10.1 mówi nam, że w poszukiwaniu kwadratur o najwyższym rzędzie możemy ograniczyć się do kwadratur interpolacyjnych. Ponadto, maksymalny rząd jest na pewno nie mniejszy niż $n+1$ i na pewno nie większy niż $2(n+1)$. Aby jednak odpowiedzieć na pytanie jaki jest rzeczywisty maksymalny rząd, musimy odwołać się do pewnych faktów z teorii aproksymacji w przestrzeniach unitarnych, a ściślej, do własności ciągów wielomianów ortogonalnych.

10.2 Ciągi wielomianów ortogonalnych

Waga ρ definiuje nam przestrzeń $\mathcal{L}_{2,\rho}(a,b)$ funkcji ciaągłych o wartościach rzeczywistych, określonych na przedziale (a,b) i takich, że całka $\int_a^b f^2(x)\rho(x)\,dx$ istnieje i jest skończona. W $\mathcal{L}_{2,\rho}(a,b)$ wprowadzimy iloczyn skalarny

$$\langle f,g\rangle \;=\; \int_a^b f(x)g(x)\rho(x)\,dx$$

i generowaną przez ten iloczyn normę

$$\|f\| \;=\; \sqrt{\langle f,f\rangle} \;=\; \sqrt{\int_a^b f^2(x)\rho(x)\,dx}.$$

W ten sposób, $\mathcal{L}_{2,\rho}(a,b)$ jest przestrzenią unitarną i można mówić o ortogonalności (prostopadłości) elementów w tej przestrzeni.

Definicja 10.2 *Ciąg wielomianów p_k, $k = 0,1,2,\ldots$, nazywamy ciągiem wielomianów ortogonalnych w przestrzeni $\mathcal{L}_{2,\rho}(a,b)$ jeśli*

(i) *wielomian p_k jest stopnia dokładnie k,*

$$deg\, p_k \;=\; k, \qquad \forall k \geq 0,$$

(ii) *wielomiany te są wzajemnie ortogonalne w przestrzeni $\mathcal{L}_{2,\rho}(a,b)$, tzn.*

$$\langle p_i,p_j\rangle \;=\; \int_a^b p_i(x)p_j(x)\rho(x)\,dx \;=\; 0, \qquad i \neq j.$$

Pokażemy, że ta definicja ma sens.

Lemat 10.2 *Ciąg wielomianów ortogonalnych $\{p_k\}_{k\geq 0}$ w przestrzeni unitarnej $\mathcal{L}_{2,\rho}(a,b)$ istnieje. Jeśli dodatkowo założymy, że $p_k(x) = x^k + \ldots$, tzn. współczynnik przy najwyższej potędze wielomianu p_k wynosi 1, to ciąg taki jest wyznaczony jednoznacznie.*

Dowód Wielomiany p_k można skonstruować stosując proces ortogonalizacji Grama-Schmidta, tzn. ortogonalizując dowolny ciąg wielomianów $\{w_k\}_{k\geq 0}$ spełniający deg $w_k = k$. Rzeczywiście, weźmy np. $w_k(x) = x^k$. Jeśli położymy $p_0 = w_0$ i dalej indukcyjnie

$$p_k \;=\; w_k \;-\; \sum_{j=0}^{k-1} \frac{\langle w_k,p_j\rangle}{\langle p_j,p_j\rangle} p_j, \qquad k \geq 1,$$

to, jak łatwo bezpośrednio sprawdzić, $\{p_k\}_{k\geq 0}$ będzie ciągiem wielomianów ortogonalnych i $p_k(x) = x^k + \ldots$.

Aby pokazać jednoznaczność załóżmy, że istnieje inny ciąg $\{q_k\}_{k\geq 0}$ wielomianów ortogonalnych, w którym współczynniki przy najwyższych

potęgach wynoszą 1. Ponieważ deg $p_k = k$, $\forall k$, wielomiany $p_0, p_1, \ldots, p_k$ tworzą bazę ortogonalną w Π_k (ze względu na iloczyn skalarny $\langle\cdot,\cdot\rangle$), a stąd

$$q_k \;=\; \sum_{j=0}^{k} \frac{\langle q_k, p_j\rangle}{\langle p_j, p_j\rangle} p_j.$$

Z drugiej strony, dla $0 \le j \le k-1$, mamy $\langle q_k, p_j\rangle = 0$, bo ortogonalność q_k do wielomianów bazowych q_j, $0 \le j \le k-1$, jest równoważna ortogonalności do całej przestrzeni Π_{k-1}, a w szczególności do p_j. Stąd

$$q_k \;=\; \frac{\langle q_k, p_k\rangle}{\langle p_k, p_k\rangle} p_k,$$

a ponieważ współczynniki przy x^k w wielomianach q_k i p_k są takie same, to $q_k = p_k$. To kończy dowód. $\square$

Pokazaliśmy, że ciąg wielomianów ortogonalnych jest wyznaczony jednoznacznie z dokładnością do współczynników przy najwyższych potęgach x^k. Podamy teraz kilka przykładów takich ciągów.

Wielomiany Legendre'a L_k.
Jest to najbardziej naturalnie zdefiniowany ciąg wielomianów ortogonalnych, bowiem w tym przypadku $[a,b] = [-1,1]$ i waga jest jednostkowa, $\rho \equiv 1$. Mamy

$$L_k(x) \;=\; \frac{1}{2^k k!}\frac{d^k}{dx^k}(x^2-1)^k,$$

albo

$$\begin{aligned} L_0(x) &= 1, \qquad L_1(x) \;=\; x, \\ L_k(x) &= \frac{2k-1}{k} x L_{k-1}(x) \;-\; \frac{k-1}{k} L_{k-2}(x), \qquad k \ge 2. \end{aligned}$$

Wielomiany Czebyszewa T_k.
Ten ciąg wielomianów poznaliśmy już w Rozdziale 7.2. Są one ortogonalne w przedziale $[-1,1]$ z wagą $\rho(x) = (1-x^2)^{-1/2}$. Przypomnijmy, że $T_k(x) = \cos(k \arccos(x))$ albo

$$\begin{aligned} T_0(x) &= 1, \qquad T_1(x) \;=\; x, \\ T_k(x) &= 2xT_{k-1}(x) \;-\; T_{k-2}(x), \qquad k \ge 2. \end{aligned}$$

Wielomiany Hermite'a H_k.
Jest to przykład wielomianów ortogonalnych na całej prostej rzeczywistej, a więc $(a,b) = (-\infty, +\infty)$, przy czym waga $\rho(x) = e^{-x^2}$. Wielomiany Hermite'a dane są wzorami

$$H_k(x) \,=\, (-1)^k e^{x^2} \frac{d^k}{dx^k} e^{-x^2},$$

albo

$$\begin{aligned} H_0(x) &= 1, \qquad H_1(x) \,=\, 2x, \\ H_k(x) &= 2xH_{k-1}(x) \,-\, (2k-2)H_{k-2}(x), \qquad k \geq 2. \end{aligned}$$

Zauważmy, że we wszystkich tych przykładach kolejny wielomian ortogonalny zależał tylko od dwóch poprzednich. Nie jest to przypadek, ale ogólna reguła. Wielomiany ortogonalne spełniają bowiem formułę trójczłonową, zob. U. 10.1.

Ciągi wielomianów ortogonalnych mają również szereg innych ciekawych własności, co powoduje, że często używane są jako narzędzie do aproksymacji funkcji wielomianami, zob. U. 10.2. Teraz pokażemy tylko jedną bardzo ważną ich własność, a dokładniej, własność zer tych wielomianów, która będzie nam potrzebna do rozwiązania problemu kwadratur maksymalnego rzędu.

Lemat 10.3 *Wielomian ortogonalny p_k ma dokładnie k pojedynczych i różnych zer w przedziale otwartym (a,b).*

Dowód Niech $y_1, y_2, \ldots, y_s$ będą wszystkimi punktami z przedziału (a,b), w których wielomian p_k zmienia znak. Gdyby lemat nie był prawdziwy to mielibyśmy $s < k$. Wtedy, kładąc

$$w(x) \,=\, (x-y_1)(x-y_2)\cdots(x-y_s),$$

mamy $\langle w, p_k \rangle = 0$, bo $w \in \Pi_s$ i $s < k$. Z drugiej strony,

$$\langle w, p_k \rangle \,=\, \int_a^b w(x) p_k(x) \rho(x)\, dx \,\neq\, 0,$$

bo funkcja podcałkowa jest albo stale dodatnia, albo stale ujemna (z wyjątkiem zbioru miary zero). Ta sprzeczność kończy dowód. $\square$

10.3 Kwadratury Gaussa

Dla dowolnej n, niech $x_0^*, x_1^*, \ldots, x_n^*$ będą (różnymi) zerami $(n+1)$-szego wielomianu p_{n+1} w ciągu wielomianów ortogonalnych w przestrzeni unitarnej $\mathcal{L}_{2,\rho}(a,b)$, tzn.

$$p_{n+1}(x) \;=\; (x - x_0^*)(x - x_1^*)\cdots(x - x_n^*). \tag{10.1}$$

Ponieważ x_j^* leżą w przedziale (a, b), możemy mówić o kwadraturze interpolacyjnej opartej na tych węzłach.

Definicja 10.3 *Kwadraturę interpolacyjną Q_n^{GS} opartą na zerach wielomianu ortogonalnego p_{n+1} nazywamy kwadraturą Gaussa.*

Okazuje się, że właśnie kwadratury Gaussa mają najwyższy rząd. Dokładniej, mamy następujące twierdzenie.

Twierdzenie 10.2 *Kwadratura Gaussa Q_n^{GS} ma najwyższy rząd spośród wszystkich kwadratur opartych na węzłach o łącznej krotności $n+1$, oraz*

$$\mathrm{rz}(Q_n^{GS}) \;=\; 2n + 2.$$

Dowód Wobec Lematu 10.1(ii) wystarczy pokazać, że kwadratura Q_n^{GS} jest dokładna dla każdego wielomianu stopnia nie większego niż $2n+1$. Niech $f \in \Pi_{2n+1}$. Niech $w_f \in \Pi_n$ będzie wielomianem interpolującym f w węzłach-zerach $x_0^*, \ldots, x_n^*$ wielomianu p_{n+1}. Jeśli $\deg f \le n$ to oczywiście $w_f = f$ i $Q_n^{GS}(f) = S_\rho(w_f) = S_\rho(f)$. Jeśli zaś

$$n + 1 \;\le\; \deg f \;\le\; 2n + 1,$$

to $f - w_f$ jest wielomianem stopnia tego samego co f i zeruje się w x_j^*, $0 \le j \le n$. Stąd

$$f(x) - w_f(x) \;=\; (x - x_0^*)(x - x_1^*)\cdots(x - x_n^*)g(x),$$

gdzie g jest wielomianem,

$$\deg g \;=\; \deg f \;-\; (n+1) \;\le\; (2n+1) \;-\; (n+1) \;=\; n.$$

Korzystając z (10.1) i faktu, że p_{n+1} jest prostopadły do Π_n, ostatecznie otrzymujemy

$$\begin{aligned} S_\rho(f) - Q_n^{GS}(f) &= \int_a^b (f(x) - w_f(x))\rho(x)\,dx \\ &= \int_a^b (x - x_0^*)\cdots(x - x_n^*)g(x)\rho(x)\,dx \\ &= \int_a^b p_{n+1}(x)g(x)\rho(x) = \langle p_{n+1}, g\rangle = 0, \end{aligned}$$

co kończy dowód. □

Zajmiemy się teraz błędem kwadratur Gaussa. Pokażemy, że również ze względu na błąd mają one dobre własności.

Twierdzenie 10.3 *Jeśli* $f \in \mathbf{C}^{2n+2}([a,b])$ *to błąd kwadratury Gaussa* Q_n^{GS} *wyraża się wzorem*

$$S_\rho(f) - Q_n^{GS}(f) = \|p_{n+1}\|^2 \frac{f^{(2n+2)}(\xi)}{(2n+2)!},$$

gdzie $\xi \in [a,b]$. *Stąd, w szczególności,*

$$\max_{f \in F_M^{2n+1}([a,b])} |S_\rho(f) - Q_n^{GS}(f)| = \frac{M\,\|p_{n+1}\|^2}{(2n+2)!}.$$

Dowód Niech $w_f \in \Pi_n$ będzie wielomianem interpolującym f w zerach x_j^* wielomianu p_{n+1}. Niech $\tilde{w}_f \in \Pi_{2n+1}$ będzie z kolei wielomianem (Hermite'a) interpolującym f w dwukrotnych węzłach x_j^*, tzn. takim, że $\tilde{w}_f(x_j^*) = f(x_j^*)$ i $\tilde{w}'_f(x_j^*) = f'(x_j^*)$, $0 \le j \le n$. Ponieważ $\mathrm{rz}(Q_n^{GS}) = 2n+2$ to $Q_n^{GS}(\tilde{w}_f) = S_\rho(\tilde{w}_f)$, a stąd i ze wzoru na błąd interpolacji Hermite'a mamy

$$\begin{aligned} S_\rho(f) - Q_n^{GS}(f) &= S_\rho(f) - Q_n^{GS}(w_f) = S_\rho(f) - Q_n^{GS}(\tilde{w}_f) \\ &= S_\rho(f) - S_\rho(\tilde{w}_f) = \int_a^b (f(x) - \tilde{w}_f(x))\rho(x)\,dx \\ &= \int_a^b (x - x_0^*)^2 \cdots (x - x_n^*)^2 f(x_0^*, \ldots, x_n^*, x)\rho(x)\,dx \\ &= \int_a^b p_{n+1}^2(x)\rho(x) f(x_0^*, \ldots, x_n^*, x)\,dx. \end{aligned}$$

Ponieważ funkcja $p_{n+1}^2(x)\rho(x)$ jest prawie wszędzie dodatnia, możemy teraz zastosować twierdzenie o wartości średniej, aby ostatecznie otrzymać

$$\begin{aligned} S_\rho(f) - Q_n^{GS}(f) &= \frac{f^{(2n+2)}(\xi)}{(2n+2)!} \int_a^b p_{n+1}^2(x)\rho(x)\,dx \\ &= \frac{f^{(2n+2)}(\xi)}{(2n+2)!} \|p_{n+1}\|^2, \end{aligned}$$

co kończy dowód. $\square$

Na końcu, zwrócimy jeszcze uwagę na inną, bardzo ważną własność kwadratur Gaussa, a mianowicie, że ich współczynniki są dodatnie. Rzeczywiście, zapisując

$$Q_n^{GS}(f) = \sum_{j=0}^{n} a_j f(x_j)$$

i podstawiając

$$f_j(x) = (x - x_0^*)^2 \cdots (x - x_{j-1}^*)^2 (x - x_{j+1}^*)^2 \cdots (x - x_n^*)^2$$

mamy, że $f_j \in \Pi_{2n}$ i f_j jest prawie wszędzie dodatnia. Stąd

$$0 < S_\rho(f_j) = Q_n^{GS}(f_j) = a_j f_j(x_j^*)$$

i $a_j > 0$, bo $f_j(x_j^*) > 0$. Przypomnijmy, że dodatniość współczynników kwadratury ma duże znaczenie przy numerycznym ich obliczaniu, zwłaszcza gdy funkcja podcałkowa f ma stały znak, zob. Rozdział 2.5.2.

Mimo niewątpliwych zalet kwadratur Gaussa, ich stosowalność ograniczają trudności w wyliczeniu pierwiastków wielomianów ortogonalnych, gdy stopień wielomianu jest duży. Wyjątkiem są tutaj kwadratury interpolacyjne oparte na zerach wielomianów Czebyszewa, zob. U. 10.3.

Uwagi i uzupełnienia

U. 10.1 Pokażemy teraz, że wielomiany ortogonalne $\{p_k\}_{k\geq 0}$ w danej przestrzeni $\mathcal{L}_{2,\rho}(a,b)$ spełniają następującą formułę trójczłonową. Załóżmy dla

uproszczenia, że współczynnik przy x^k w wielomianie p_k jest dla każdego k jednością. Wtedy istnieją liczby β_k (dla $k \geq 1$) i $\gamma_k > 0$ (dla $k \geq 2$) takie, że

$$\begin{aligned} p_0(x) &= 1, \\ p_1(x) &= (x - \beta_1), \\ p_k(x) &= (x - \beta_k)p_{k-1}(x) - \gamma_k p_{k-2}(x), \qquad k \geq 2. \end{aligned} \tag{10.2}$$

Aby to pokazać zauważmy, że p_k można dla $k \geq 1$ przedstawić w postaci rozwinięcia

$$p_k(x) = (x - c_{k-1})p_{k-1}(x) + \sum_{j=0}^{k-2} c_j p_j(x).$$

Mnożąc skalarnie obie strony tego równania przez p_s dla $0 \leq s \leq k-3$, otrzymujemy

$$0 = \langle p_k(x), p_s(x) \rangle = \langle (x - c_{k-1})p_{k-1}(x), p_s(x) \rangle + c_s \langle p_s(x), p_s(x) \rangle.$$

Wobec tego, że $(x - c_{k-1})p_s(x)$ jest wielomianem stopnia mniejszego niż $k-1$, mamy

$$\langle (x - c_{k-1})p_{k-1}(x), p_s(x) \rangle = \langle p_{k-1}(x), (x - c_{k-1})p_s(x) \rangle = 0,$$

a stąd $c_s \langle p_s(x), p_s(x) \rangle = 0$ i $c_s = 0$. Możemy więc napisać, że dla $k = 1$ mamy $p_1(x) = (x - \beta_1)$, a dla $k \geq 2$,

$$p_k(x) = (x - \beta_k)p_{k-1}(x) - \gamma_k p_{k-2}(x), \tag{10.3}$$

gdzie $\beta_k = c_{k-1}$ i $\gamma_k = c_{k-2}$. Aby jawnie wyznaczyć β_k i γ_k, pomnożymy skalarnie obie strony równania (10.3) kolejno przez p_{k-1} i p_{k-2}. Otrzymujemy

$$\begin{aligned} 0 &= \langle p_k(x), p_{k-1} \rangle \\ &= \langle (x - \beta_k)p_{k-1}(x), p_{k-1}(x) \rangle - \gamma_k \langle p_{k-2}(x), p_{k-1}(x) \rangle \\ &= \langle x p_{k-1}(x), p_{k-1}(x) \rangle - \beta_k \langle p_{k-1}(x), p_{k-1}(x) \rangle, \end{aligned}$$

czyli

$$\beta_k = \frac{\langle x p_{k-1}(x), p_{k-1}(x) \rangle}{\langle p_{k-1}(x), p_{k-1}(x) \rangle},$$

oraz

$$\begin{aligned} 0 &= \langle p_k(x), p_{k-2}(x) \rangle \\ &= \langle (x - \beta_k)p_{k-1}(x), p_{k-2}(x) \rangle - \gamma_k \langle p_{k-2}(x), p_{k-2}(x) \rangle \\ &= \langle p_{k-1}(x), x p_{k-2}(x) \rangle - \gamma_k \langle p_{k-2}(x), p_{k-2}(x) \rangle, \end{aligned}$$

a stąd i z równości $\langle p_{k-1}(x), xp_{k-2}(x)\rangle = \langle p_{k-1}(x), p_{k-1}(x)\rangle$,

$$\gamma_k = \frac{\langle p_{k-1}(x), p_{k-1}(x)\rangle}{\langle p_{k-2}(x), p_{k-2}(x)\rangle}.$$

Zauważmy, że z formuły trójczłonowej wynika w szczególności algorytm wyznaczenia ciągu wielomianów ortogonalnych. Wystarczy bowiem wyznaczać kolejne współczynniki β_k i γ_k (obliczając odpowiednie iloczyny skalarne $\langle xp_k(x), p_k(x)\rangle$ i $\langle p_k(x), p_k(x)\rangle$) i stosować wzór rekurencyjny (10.2). Dodajmy jeszcze, że w obliczeniach numerycznych najlepiej jest przechowywać informację o ciągu $\{p_k\}_{k\geq 0}$ po prostu w postaci liczb β_k i γ_k.

U. 10.2 Wygodnie jest posłużyć się wielomianami ortogonalnymi w przypadku, gdy chcemy znaleźć najlepszą aproksymację danej funkcji f wielomianem ustalonego stopnia n, i błąd mierzymy w normie przestrzeni $\mathcal{L}_{2,\rho}(a,b)$. Rzeczywiście, jak wiadomo, najlepszą aproksymacją dla f w przestrzeni Π_n jest jej rzut prostopadły na Π_n. Ponieważ $n+1$ początkowych wielomianów ortogonalnych p_k, $0 \leq k \leq n$, tworzy bazę w Π_n, rzut ten wyraża się wzorem

$$w_f^* = \sum_{k=0}^{n} \frac{\langle f, p_k\rangle}{\langle p_k, p_k\rangle} p_k.$$

U. 10.3 Zachodzi następujące twierdzenie Łuzina. Niech przedział $[a,b]$ będzie skończony. Niech $Q_n(f) = \sum_{j=0}^{n} a_j^{(n)} f(x_j^{(n)})$ będzie takim ciągiem kwadratur, że:

(i) wszystkie współczynniki $a_j^{(n)}$ są dodatnie,

(ii) rząd kwadratur Q_n rośnie do nieskończoności gdy $n \to \infty$.

Wtedy ciąg $Q_n(f)$ zbiega do $\int_a^b f(x)dx$ dla każdej funkcji ciągłej f.

Zauważmy, że twierdzenie to stosuje się do ciągu kwadratur Gaussa Q_n^{GS}, ale nie do ciągu kwadratur Newtona-Cotesa Q_m^{NC}, ponieważ w tych ostatnich pojawiają się dla dużych n współczynniki ujemne.

W praktyce, najczęściej stosuje się ciąg kwadratur interpolacyjnych opartych na zerach kolejnych wielomianów Czebyszewa, ponieważ zera te dane są jawnie i "zagęszczają się", tzn. zera wielomianu Czebyszewa T_k są też zerami wielomianu T_{2k}. Powstające w ten sposób kwadratury noszą nazwę *formuł Clanshow-Curtis'a*. Są one w pewnym sensie uniwersalne, bowiem posiadają optymalną szybkość zbieżności $n^{-(r+1)}$ w klasach $F_M^r([a,b])$ dla dowolnych r i M.

U. 10.4 Jeśli przedział całkowania jest skończony i waga jest jednostkowa to kwadratury Gaussa (a dokładniej kwadratury Legendre'a) można użyć do tworzenia kwadratur złożonych $\bar{Q}^{GS}_{r,k}$, gdzie k oznacza liczbę podprzedziałów. Łatwo widać, że dla $f \in F^{2r+1}_M([a,b])$, błąd takiej kwadratury można oszacować przez

$$|S(f) \,-\, \bar{Q}^{GS}_{r,k}| \,\le\, \frac{M(b-a)^{2r+3}}{(2r+2)!}\Big(\frac{1}{k}\Big)^{2r+2},$$

czyli jest on porównywalny do błędu "zwykłej" złożonej kwadratury interpolacyjnej. Jednak złożona kwadratura Legendre'a korzysta z dwa razy mniejszej liczby węzłów.

U. 10.5 Błąd złożonej kwadratury Legendre'a w klasie $F^{2r+1}_M([a,b])$ można podać dokładnie. Wystarczy wykorzystać wzory z Ćw. 10.4 i 10.5, aby otrzymać

$$\max_{f\in F^{2r+1}_M([a,b])} |S(f) - \bar{Q}^{GS}_{r,k}| \,=\, C\,\Big(\frac{1}{n}\Big)^{2r+2},$$

gdzie

$$C \,=\, \frac{M2^{2r+3}(r+1)^{2r+2}}{(2r+3)!}\Big(\frac{(r+2)\cdots(2r+2)}{1\cdot 2\cdots(r+1)}\Big)^2$$

i $n = k(r+1)$ jest ogólną liczbą węzłów na $[a,b]$.

Ćwiczenia

Ćw. 10.1 Uzasadnić, że:
(a) jeśli kwadratura jest dokładna dla dowolnych $n+1$ wielomianów tworzących bazę w Π_n, to jest ona rzędu co najmniej $n+1$, oraz
(b) jeśli kwadratura jest rzędu $n+1$ to jest ona niedokładna dla każdego wielomianu stopnia dokładnie $n+1$.

Ćw. 10.2 Uzasadnić, że kwadratura prostokątów jest kwadraturą Legendre'a, natomiast żadna z kwadratur Newtona-Cotesa Q^{NC}_n nie jest kwadraturą Gaussa.

Ćw. 10.3 Załóżmy, że dane są liczby β_k i γ_k definiujące ciąg wielomianów ortogonalnych przez formułę trójczłonową. Zaproponować ekonomiczny (tzn. o koszcie proporcjonalnym do n) algorytm obliczania wartości n-tego wielomianu ortogonalnego w danym punkcie x, wykorzystujący formułę trójczłonową.

Ćw. 10.4 Niech x_j, $0 \le j \le n$, będą zerami $(n+1)$-szego wielomianu Legendre'a. Pokazać, że

$$\int_{-1}^{1} (x-x_0)^2(x-x_1)^2\cdots(x-x_n)^2\,dx$$
$$= \frac{2^{2n+3}}{2n+3}\Big(\frac{1\cdot 2\cdots n\cdot(n+1)}{(n+2)(n+3)\cdots(2n+2)}\Big)^2.$$

Wskazówka. Wykorzystać fakt, że dla n-tego wielomianu Legendre'a mamy $\int_{-1}^{1} L_n^2(x)dx = (2n+1)^{-1}$.

Ćw. 10.5 Niech

$$Q_n^{GS}(f) \,=\, \sum_{j=0}^{n} a_j f(x_j)$$

będzie kwadraturą interpolacyjną opartą na zerach $(n+1)$-ezego wielomianu Legendre'a. Niech $-\infty < a < b < \infty$. Pokazać, że wtedy kwadratura

$$\tilde{Q}_n^{GS}(f) \,=\, \frac{b-a}{2}\sum_{j=0}^{n} f\Big(a + \frac{x_j+1}{2}(b-a)\Big)$$

jest kwadraturą Gaussa opartą na $n+1$ węzłach, dla całki na przedziale (a,b) z wagą jednostkową. Ponadto, jeśli $f \in \mathbf{C}^{(2n+2)}([a,b])$, to

$$\int_a^b f(x)\,dx \,-\, \tilde{Q}_n^{GS}(f) \,=\, \Big(\frac{b-a}{2}\Big)^{2n+3}\Big(\frac{1\cdot 2\cdots(n+1)}{(n+2)\cdots(2n+2)}\Big)^2\frac{f^{(2n+2)}(\xi)}{(2n+3)!}.$$

Ćw. 10.6 Niech $\rho \equiv 1$ i $-\infty < a < b < +\infty$. Pokazać, że jeśli kwadratura Q oparta na $n+1$ węzłach jest rzędu $r \ge n+1$, to odpowiadająca jej kwadratura złożona $\bar{Q}_k$ ma następującą własność. Jeśli $f \in \mathbf{C}^{(r)}([a,b])$ to

$$|S(f) \,-\, \bar{Q}_k(f)| \,\le\, \Big(\frac{1}{k}\Big)^{r-1}\frac{(b-a)^r}{r!}\|f^{(r)}\|_{\mathbf{C}([a,b])}.$$

Ćw. 10.7 Niech x_j, $0 \le j \le n$, będą zerami $(n+1)$-szego wielomianu ortogonalnego Legendre'a (tzn. na przedziale $[-1,1]$ z wagą 1). Niech dla $0 \le j \le n$,

$$w_j \,=\, \int_{-1}^{1} \frac{(x-x_0)\cdots(x-x_{j-1})(x-x_{j+1})\cdots(x-x_n)}{(x_j-x_0)\cdots(x_j-x_{j-1})(x_j-x_{j+1})\cdots(x_j-x_n)}\,dx.$$

Pokazać, że jeśli f i g są wielomianami stopnia nie większego niż n, to ich iloczyn skalarny w $\mathcal{L}_{2,1}(-1,1)$,

$$\langle f,g\rangle \,=\, \int_{-1}^{1} f(x)g(x)\,dx \,=\, \sum_{j=0}^{n} w_j f(x_j)g(x_j).$$

Rozdział 11

Iteracje dla równań liniowych

Algorytmy rozwiązywania układów równań liniowych postaci

$$A\vec{x} = \vec{b},$$

gdzie A jest nieosobliwą macierzą rzeczywistą $n \times n$, a $\vec{b}$ jest wektorem rzeczywistym w $\mathbf{R}^n$, które rozpatrywaliśmy w Rozdziałach 3, 4 i 5, należą do grupy algorytmów *dokładnych* albo *bezpośrednich*. To znaczy, że po wykonaniu skończonej liczby dopuszczalnych operacji elementarnych dostajemy w arytmetyce idealnej dokładne rozwiązanie

$$\vec{x}^* = A^{-1}\vec{b}.$$

W tym rozdziale zajmiemy się *algorytmami iteracyjnymi* rozwiązywania układów równań liniowych. Polegają one na tym, że, startując z pewnego przybliżenia początkowego $\vec{x}_0$, konstruuje się ciąg kolejnych przybliżeń

$$\vec{x}_k = \Phi_k(A, \vec{b}; \vec{x}_0), \qquad k = 1, 2, \ldots,$$

które w granicy osiągają rozwiązanie dokładne,

$$\lim_{k\to\infty} \vec{x}_k = \vec{x}^*.$$

11.1 Kiedy stosujemy iteracje?

Jasne jest, że algorytmy iteracyjne stosujemy wtedy, gdy są one konkurencyjne w stosunku do algorytmów bezpośrednich. Dlatego przekształcenia Φ_k należy wybierać tak, aby kolejne przybliżenia można było łatwo obliczać i jednocześnie kolejne błędy $\|\vec{x}_k - \vec{x}^*\|$ szybko zbiegały do zera.

Zwykle zakłada się również, że dokładne rozwiązanie $\vec{x}^*$ jest punktem stałym przekształcenia $\Phi_k(A, \vec{b}; \cdot)$. Wtedy kolejne błędy spełniają zależność

$$\vec{x}_k - \vec{x}^* \;=\; \Phi_k(A, \vec{b}; \vec{x}_0) - \Phi_k(A, \vec{b}; \vec{x}^*).$$

Jeśli teraz $\Phi_k(A, \vec{b}; \cdot)$ są lipschitzowskie ze stałymi $m_k < +\infty$, tzn. dla pewnej normy wektorowej $\|\cdot\|$ mamy

$$\|\Phi_k(\vec{x}) - \Phi_k(\vec{y})\| \;\le\; m_k\,\|\vec{x} - \vec{y}\|, \qquad \forall \vec{x}, \vec{y},$$

to

$$\|\vec{x}_k - \vec{x}^*\| \;\le\; m_k \|\vec{x}_0 - \vec{x}^*\|.$$

Warunek $\lim_{k\to\infty} m_k = 0$ jest więc dostateczny na to, aby metoda była zbieżna dla dowolnego przybliżenia początkowego $\vec{x}_0$, przy czym szybkość zbieżności zależy od tego, jak szybko m_k maleją do zera. Dla większości stosowanych metod Φ_k jest funkcją liniową błędu początkowego, tzn.

$$\Phi_k(A, \vec{b}; \vec{x}_0 - \vec{x}^*) \;=\; M_k(\vec{x}_0 - \vec{x}^*),$$

gdzie M_k jest pewną macierzą. Wtedy jako m_k można przyjąć normę tej macierzy,

$$m_k \;=\; \|M_k\| \;=\; \sup_{\|\vec{x}\|=1} \|M_k \vec{x}\|.$$

Dla ilustracji, rozpatrzmy ogólną metodę *iteracji prostej*, w której

$$\vec{x}_k \;=\; B\vec{x}_{k-1} \,+\, \vec{c}, \tag{11.1}$$

dla pewnej macierzy B wymiaru $n \times n$ i wektora $\vec{c} \in \mathbf{R}^n$. W tym przypadku

$$\vec{x}_k - \vec{x}^* \;=\; B^k(\vec{x}_0 - \vec{x}^*),$$

a stąd i z nierówności $\|B^k\| \leq \|B\|^k$, mamy

$$\|\vec{x}_k - \vec{x}^*\| \;\leq\; \|B\|^k \|\vec{x}_0 - \vec{x}^*\|.$$

Warunkiem dostatecznym zbieżności iteracji prostych jest więc $\|B\| < 1$. Mówimy, że metoda jest zbieżna liniowo z ilorazem $\|B\|$.

Przykład 11.1 Rozkładając macierz $A = (a_{i,j})_{i,j=1}^n$ na sumę

$$A \;=\; D + C,$$

gdzie D jest macierzą diagonalną składającą się z wyrazów stojących na głównej przekątnej macierzy A, układ $A\vec{x} = \vec{b}$ jest równoważny układowi

$$D\vec{x} \;=\; -C\vec{x} \,+\, \vec{b},$$

a stąd (o ile na przekątnej macierzy A nie mamy zera) otrzymujemy metodę iteracyjną

$$\vec{x}_k \;=\; B\vec{x}_{k-1} \,+\, \vec{c},$$

gdzie $B = -D^{-1}C$ i $\vec{c} = D^{-1}\vec{b}$, zwaną *metodą Jacobiego.*

W metodzie Jacobiego warunek dostateczny zbieżności, $\|B\| < 1$, jest spełniony wtedy, gdy macierz A ma dominującą przekątną, tzn. gdy

$$2|a_{i,i}| \;>\; \sum_{j=1}^n |a_{i,j}|, \qquad 1 \leq i \leq n. \tag{11.2}$$

Rzeczywiście, ponieważ wyraz (i,j) macierzy $D^{-1}C$ wynosi 0 dla $i = j$ i $a_{i,j}/a_{i,i}$ dla $i \neq j$, to

$$\begin{aligned} \|D^{-1}C\|_\infty \;&=\; \max_{1\leq i\leq n} \sum_{j=1,j\neq i}^n |a_{i,j}|/|a_{i,i}| \\ &=\; \max_{1\leq i\leq n} \sum_{j=1}^n |a_{i,j}|/|a_{i,i}| \,-\, 1 \;<\; 1, \end{aligned}$$

przy czym ostatnia nierówność wynika z (11.2).

Inne przykłady iteracji prostych podane są w U. 11.3 i Ćw. 11.2.

Zastanówmy się teraz nad złożonością metod iteracyjnych. Ponieważ możemy jedynie znaleźć pewne przybliżenie rozwiązania dokładnego $\vec{x}^*$,

przez złożoność metody będziemy rozumieli koszt kombinatoryczny obliczenia $\vec{x}_k$ z zadaną dokładnością $\varepsilon > 0$. Dla uproszczenia założymy, że medoda jest zbieżna liniowo z ilorazem m. Zauważmy, że aby zredukować błąd początkowy do $\varepsilon > 0$, wystarczy wykonać $k = k(\varepsilon)$ iteracji, gdzie k spełnia

$$m^k \|\vec{x}_0 - \vec{x}^*\| \leq \varepsilon,$$

czyli

$$k \geq \frac{\log(1/\varepsilon) - \log(1/\|\vec{x}_0 - \vec{x}^*\|)}{\log(1/m)}.$$

Liczba ta zależy więc w istotny sposób od błędu początkowego i (przede wszystkim) od stałej Lipschitza m, natomiast zależność od dokładności ε i wymiaru n układu jest dużo mniej istotna. Zakładając, że koszt jednej iteracji wynosi $c = c(n)$ (zwykle $c(n)$ jest tym mniejszy, im mniejsza jest liczba niezerowych elementów macierzy A), złożoność metody jest proporcjonalna do

$$c(n)\,\frac{\log(1/\varepsilon)}{\log(1/m)}.$$

Stąd oczywisty wniosek, że metody iteracyjne warto stosować zamiast metod bezpośrednich w przypadku gdy

- wymiar n układu $A\vec{x} = \vec{b}$ jest "duży", oraz
- macierz A układu jest "rozrzedzona", tzn. ma stosunkowo niewielką liczbę elementów niezerowych, np. proporcjonalną do n.

Układy o tych własnościach powstają często przy numerycznym rozwiązywaniu równań różniczkowych cząstkowych.

Zaletą metod iteracyjnych jest również ich prostota, przez co są one łatwe do zaprogramowania.

11.2 Metoda Czebyszewa

Zauważyliśmy, że ze względu na szybkość zbieżności metody iteracyjnej ważne jest, aby stałe lipschitzowskie m_k odwzorowań Φ_k malały jak najszybciej. *Metoda Czebyszewa*, którą przedstawimy w tym rozdziale, jest właśnie próbą minimalizacji tych stałych.

Przy opisie metody będziemy korzystać z następujących dwóch faktów, które sformułujemy jako lematy.

Lemat 11.1 *Jeśli macierz A jest symetryczna, to istnieje ortonormalna w $\mathbf{R}^n$ baza jej wektorów własnych $\vec{\xi}_j$, $1 \le j \le n$, tzn.*

$$\langle \vec{\xi}_i, \vec{\xi}_j \rangle_2 \,=\, \vec{\xi}_i^T \vec{\xi}_j \,=\, \begin{cases} 0 & i \neq j, \\ 1 & i = j, \end{cases}$$

a odpowiadające im wartości własne λ_j, $A\vec{\xi}_j = \lambda_j \vec{\xi}_j$, są rzeczywiste. Jeśli ponadto A jest dodatnio określona to λ_j są dodatnie.

Lemat 11.2 *Niech macierz A będzie symetryczna i niech*

$$|\lambda_1| \ge |\lambda_2| \ge \cdots \ge |\lambda_n|$$

będą jej wartościami własnymi. Wtedy

$$\|A\|_2 \,=\, \sup_{\vec{x} \neq \vec{0}} \frac{\|A\vec{x}\|_2}{\|\vec{x}\|_2} \,=\, |\lambda_1|.$$

Jeśli ponadto A jest nieosobliwa, tzn. $|\lambda_n| > 0$, to

$$\|A^{-1}\|_2 \,=\, \frac{1}{|\lambda_n|}.$$

Lemat 11.1 jest znanym faktem z algebry liniowej, więc dowód pominiemy. Dowód Lematu 11.2 podany jest w U. 11.1.

Zakładamy, że macierz A wyjściowego układu równań jest symetryczna i dodatnio określona.

$$A \,=\, A^T \,>\, 0,$$

a jej wartości własne λ_j leżą w znanym przedziale $[a, b]$,

$$0 < a \le \lambda_n \le \cdots \le \lambda_1 = \|A\|_2 \le b < \infty.$$

W metodzie Czebyszewa kolejne przybliżenia $\vec{x}_k$ konstruujemy tak, aby była spełniona równość

$$(\vec{x}_k - \vec{x}^*) \,=\, W_k(A)(\vec{x}_0 - \vec{x}^*), \tag{11.3}$$

gdzie $W_k(A)$ jest macierzą, która wielomianowo zależy od A,

$$W_k(A) \,=\, \sum_{j=0}^{k} a_j^{(k)} A^k,$$

i ma możliwie małą normę. (Dalej będziemy również używać tego samego symbolu W_k do oznaczenia wielomianu zmiennej rzeczywistej, tzn. dla $t \in \mathbf{R}$ mamy $W_k(t) = \sum_{j=0}^{k} a_j^{(k)} t^k$.) Możemy więc napisać, że

$$\vec{x}_k \,=\, W_k(A)\vec{x}_0 \,+\, (I - W_k(A))\vec{x}^*.$$

Aby zapewnić konstruowalność przybliżenia $\vec{x}_k$, musimy umieć obliczać $(I - W_k(A))\vec{x}^*$. Ponieważ

$$\begin{aligned} (I - W_k(A))\vec{x}^* \;&=\; -\sum_{j=1}^{k} a_j^{(k)} A^j \vec{x}^* \,+\, (1 - a_0^{(k)})\vec{x}^* \\ &=\; -\sum_{j=0}^{k-1} a_{j+1}^{(k)} A^j \vec{b} \,+\, (1 - a_0^{(k)})\vec{x}^*, \end{aligned}$$

dla konstruowalności $\vec{x}_k$ wystarczy założyć, że $a_0^{(k)} = 1$, co odpowiada warunkowi

$$W_k(0) \,=\, 1, \qquad \forall k. \tag{11.4}$$

Z nierówności (11.3) mamy

$$\|\vec{x}_k - \vec{x}^*\|_2 \,\le\, \|W_k(A)\|_2 \|\vec{x}_0 - \vec{x}^*\|_2.$$

Idea metody polega teraz na wyborze wielomianów W_k tak, aby miały one jak najmniejszą normę $\|W_k(A)\|_2$. W tym celu zauważmy, że macierz $W_k(A)$ jest symetryczna, a jej wartości własne wynoszą $W_k(\lambda_j)$, gdzie λ_j są wartościami własnymi macierzy A, zob. Ćw. 11.1. Stąd

$$\|W_k(A)\|_2 \,\le\, \max_{a \le t \le b} |W_k(t)| \,=\, \|W_k\|_{\mathbf{C}([a,b])}.$$

Rozwiązanie problemu minimalizacji ostatniego maksimum ze względu na wszystkie wielomiany $W_k \in \Pi_k$ takie, że $W_k(0) = 1$, jest podane w Rozdziale 7 (zob. Ćw. 7.8). Przypomnijmy, że optymalny wielomian dany jest wzorem

$$W_k^*(t) \,=\, \frac{T_k(h(t))}{T_k(h(0))},$$

gdzie T_k jest k-tym wielomianem Czebyszewa, a

$$h(t) \;=\; \frac{2t-(b+a)}{b-a}.$$

Przypomnijmy również, że wielomiany Czebyszewa spełniają formułę trójczłonową, $T_0(t)=1$, $T_1(t)=t$, oraz $T_k(t)=2tT_{k-1}(t)-T_{k-2}$ dla $k\geq 2$. Pozwala to na rekurencyjną konstrukcję kolejnych $\vec{x}_k$ w zależności od $\vec{x}_{k-1}$ i $\vec{x}_{k-2}$. Dokładniej, dla pierwszego przybliżenia mamy

$$W_1^*(A) \;=\; \frac{T_1(h(A))}{T_1(h(0))} \;=\; \frac{\left(\frac{2A-(b+a)I}{b-a}\right)}{\left(-\frac{b+a}{b-a}\right)} \;=\; I \;-\; \frac{2}{b+a}A,$$

a stąd

$$\begin{aligned}
\vec{x}_1 &= W_1^*(A)\vec{x}_0 \;+\; (I-W_1(A))\vec{x}^* \;=\; \vec{x}_0 \;+\; \frac{2}{b+a}(\vec{b}-A\vec{x}_0) \\
&= \vec{x}_0 \;+\; \frac{2}{b+a}\vec{r}_0,
\end{aligned}$$

gdzie $\vec{r}_0=\vec{b}-A\vec{x}_0$ jest początkowym residuum.

Dla $k\geq 2$ wykorzystamy formułę rekurencyjną dla wielomianów Czebyszewa. Oznaczając $t_j=T_j(h(0))$, $j\geq 0$, mamy

$$\begin{aligned}
W_k^*(A) &= \frac{T_k(h(A))}{t_k} \;=\; \frac{2h(A)T_{k-1}(h(A))-T_{k-2}(h(A))}{t_k} \\
&= 2\frac{t_{k-1}}{t_k}h(A)\frac{T_{k-1}(h(A))}{t_{k-1}} \;-\; \frac{t_{k-2}}{t_k}\frac{T_{k-2}(h(A))}{t_{k-2}} \\
&= 2\frac{t_{k-1}}{t_k}h(A)W_{k-1}^*(A) \;-\; \frac{t_{k-2}}{t_k}W_{k-2}^*(A).
\end{aligned}$$

Stąd

$$\begin{aligned}
\vec{x}_k-\vec{x}^* &= W_k^*(A)(\vec{x}_0-\vec{x}^*) \\
&= 2\frac{t_{k-1}}{t_k}h(A)W_{k-1}^*(\vec{x}_0-\vec{x}^*) \;-\; \frac{t_{k-2}}{t_k}W_{k-2}^*(A)(\vec{x}_0-\vec{x}^*) \\
&= 2\frac{t_{k-1}}{t_k}h(A)(\vec{x}_{k-1}-\vec{x}^*) \;-\; \frac{t_{k-2}}{t_k}(\vec{x}_{k-2}-\vec{x}^*)
\end{aligned}$$

$$
\begin{aligned}
&= 2\frac{t_{k-1}}{t_k}\Big(\frac{2}{b-a}A - \frac{b+a}{b-a}I\Big)(\vec{x}_{k-1} - \vec{x}^*) \;-\; \frac{t_{k-2}}{t_k}(\vec{x}_{k-2} - \vec{x}^*) \\
&= -\frac{4}{b-a}\,\frac{t_{k-1}}{t_k}\vec{r}_{k-1} \;-\; 2\frac{t_{k-1}}{t_k}\frac{b+a}{b-a}\vec{x}_{k-1} \;-\; \frac{t_{k-2}}{t_k}\vec{x}_{k-2} \\
&\qquad\qquad + \Big(2\frac{t_{k-1}}{t_k}\frac{b+a}{b-a} + \frac{t_{k-2}}{t_k}\Big)\vec{x}^*.
\end{aligned}
$$

Wykorzystując wynikającą z formuły trójczłonowej równość

$$
t_k \;=\; -2\,\frac{b-a}{b+a}\,t_{k-1} - t_{k-2}
$$

otrzymujemy ostatecznie

$$
\vec{x}_k \;=\; (1+\alpha_k)\vec{x}_{k-1} \;-\; \alpha_k\vec{x}_{k-2} \;-\; \beta_k\vec{r}_{k-1},
$$

gdzie $\alpha_k = t_{k-2}/t_k$ i $\beta_k = (4t_{k-1})/(t_k(b+a))$. Algorytm wynikający z metody Czebyszewa można więc zapisać następująco.

> $\vec{x}_0 := \{dowolne\};\qquad \vec{r}_0 := \vec{b} - A\vec{x}_0;$
> $\vec{x}_1 := \vec{x}_0 + 2\vec{r}_0/(b+a);\qquad \vec{r}_1 := \vec{b} - A\vec{x}_1;$
> $t_0 := 1;\qquad t_1 := -(b+a)/(b-a);$
> $\beta := -4/(b-a);$
> **for** $j = 2, 3, \ldots$ **do**
> **begin**
> $t_k := 2t_1t_{k-1} - t_{k-2};$
> $\alpha_k := t_{k-2}/t_k;$
> $\vec{x}_k := (1+\alpha_k)\vec{x}_{k-1} - \alpha_k\vec{x}_{k-2} + \beta(t_{k-1}/t_k)\vec{r}_{k-1};$
> $\vec{r}_k := \vec{b} - A\vec{x}_k$
> **end**.

Zauważmy, że opisana metoda Czebyszewa jest *metodą dwupunktową*, bowiem do konstrukcji kolejnego przybliżenia $\vec{x}_k$ wykorzystuje się dwa poprzednie przybliżenia $\vec{x}_{k-1}$ i $\vec{x}_{k-2}$.

Zastanowimy się jeszcze nad szybkością zbieżności metody Czebyszewa. Wobec

$$
\|W_k(A)\|_2 \;=\; \frac{\|T_k(h(\cdot))\|_{\mathbf{C}([a,b])}}{|T_k(h(0))|} \;=\; \frac{1}{T_k\Big(\frac{b+a}{b-a}\Big)}
$$

mamy

$$\|\vec{x}_k - \vec{x}^*\|_2 \le \frac{\|\vec{x}_0 - \vec{x}^*\|}{T_k\left(\frac{b+a}{b-a}\right)}.$$

Aby oszacować $T_k^{-1}((b+a)/(b-a))$, wykorzystamy jawną formułę na $T_k(t)$ z Ćw. 7.2. Oznaczając $t = (b+a)/(b-a)$ mamy dla dużych k

$$\begin{aligned}
\frac{1}{T_k(t)} &= \frac{2}{(t+\sqrt{t^2-1})^k + (t-\sqrt{t^2-1})^k} \approx \frac{2}{(t+\sqrt{t^2-1})^k} \\
&= \frac{2}{\left(\frac{b-a}{b+a} + \sqrt{\frac{(b+a)^2-(b-a)^2}{(b-a)^2}}\right)^k} \\
&= \frac{2(b-a)^k}{(\sqrt{b}+\sqrt{a})^{2k}} = 2\left(\frac{\sqrt{b}-\sqrt{a}}{\sqrt{b}+\sqrt{a}}\right)^k,
\end{aligned}$$

a stąd w przybliżeniu

$$\|\vec{x}_k - \vec{x}^*\|_2 \le 2\left(\frac{\sqrt{b/a}-1}{\sqrt{b/a}+1}\right)^k \|\vec{x}_0 - \vec{x}^*\|_2.$$

Metoda Czebyszewa jest więc zbieżna co najmniej liniowo z ilorazem $(\sqrt{b/a}-1)/(\sqrt{b/a}+1)$. Zauważmy jeszcze, że jeśli a i b są odpowiednio najmniejszą i największą wartością własną macierzy A, to $b = \|A\|_2$, $1/a = \|A^{-1}\|_2$, a stąd $b/a = \|A\|_2\|A^{-1}\|_2 = \mathrm{cond}(A)$ i metoda jest zbieżna z ilorazem

$$\frac{\sqrt{\mathrm{cond}(A)}-1}{\sqrt{\mathrm{cond}(A)}+1}.$$

Po raz kolejny widzimy tu ważną rolę uwarunkowania macierzy. Im uwarunkowanie jest gorsze tym gorsza zbieżność metody Czebyszewa.

11.3 Metoda najszybszego spadku

Tak jak poprzednio, będziemy zakładać, że macierz A układu jest symetryczna i dodatnio określona. Wtedy zachodzi następujący fakt.

Lemat 11.3 *Rozwiązanie $\vec{x}^*$ układu $A\vec{x} = \vec{b}$ jest jedynym wektorem minimalizującym formę kwadratową*

$$Q(\vec{x}) \,=\, \frac{1}{2}\vec{x}^T A\vec{x} \,-\, \vec{x}^T\vec{b}.$$

Dowód Zauważmy, że dla dowolnych $\vec{x}$ i $\vec{z}$ mamy

$$\begin{aligned} Q(\vec{x}+\vec{z}) &= \frac{1}{2}(\vec{x}^T+\vec{z}^T)A(\vec{x}+\vec{z}) - (\vec{x}^T+\vec{z}^T)\vec{b} \\ &= \frac{1}{2}\vec{x}^T A\vec{x} - \vec{x}^T\vec{b} + \frac{1}{2}(\vec{z}^T A\vec{x} + \vec{x}^T A\vec{z}) - \vec{z}^T\vec{b} + \frac{1}{2}\vec{z}^T A\vec{z} \\ &= Q(\vec{x}) \,-\, z^T(\vec{b} - \vec{A}\vec{x}) \,+\, \frac{1}{2}\vec{z}^T A\vec{z}, \end{aligned}$$

przy czym w ostatniej równości wykorzystaliśmy symetrię macierzy A. Stąd dla $\vec{x} = \vec{x}^*$ i $\vec{z} \neq \vec{0}$ dostajemy

$$Q(\vec{x}^* + \vec{z}) \,=\, Q(\vec{x}^*) \,+\, \frac{1}{2}\vec{z}^T A\vec{z} \,>\, Q(\vec{x}^*),$$

co z kolei wynika z dodatniej określoności A. $\square$

W *metodzie najszybszego spadku* konstruuje się kolejne przybliżenie $\vec{x}_k$ przesuwając się od $\vec{x}_{k-1}$ w kierunku przeciwnym do gradientu formy Q tak długo jak długo wartość tej formy maleje. (Przypomnijmy, że minus gradient wyznacza kierunek lokalnie najszybszego spadku.) Ponieważ gradient Q w punkcie $\vec{x}_{k-1}$ wynosi

$$\left(\frac{\partial Q}{\partial x_i}(\vec{x}_{k-1})\right)_{i=1}^{n} \,=\, A\vec{x}_{k-1} - \vec{b} \,=\, -\vec{r}_{k-1},$$

metoda ma postać

$$\vec{x}_k \,=\, \vec{x}_{k-1} \,+\, \gamma_k\vec{r}_{k-1}, \tag{11.5}$$

gdzie γ_k jest dobrane tak, aby zminimalizować $Q(\vec{x}_k)$. Jak łatwo się przekonać,

$$\begin{aligned} Q(\vec{x}_k) &= Q(\vec{x}_{k-1} + \gamma\vec{r}_{k-1}) \\ &= Q(\vec{x}_{k-1}) + \frac{1}{2}\gamma^2\vec{r}_{k-1}^T A\vec{r}_{k-1} + \gamma\vec{r}_{k-1}^T A\vec{x}_{k-1} - \gamma\vec{r}_{k-1}^T\vec{b} \\ &= Q(x_{k-1}) + \frac{1}{2}\gamma^2\vec{r}_{k-1}^T A\vec{r}_{k-1} - \gamma\vec{r}_{k-1}^T\vec{r}_{k-1} \end{aligned}$$

jest jako funkcja γ parabolą, która przyjmuje minimum w

$$\gamma_k \;=\; \frac{\vec{r}_{k-1}^T\vec{r}_{k-1}}{\vec{r}_{k-1}^T A\vec{r}_{k-1}}. \tag{11.6}$$

Wzory (11.5) i (11.6) definiują już w pełni metodę najszybszego spadku.

Pokażemy jeszcze zbieżność metody. W tym celu zauważmy, że

$$Q(\vec{x}_k) \;=\; Q(\vec{x}_{k-1}) \;-\; \frac{1}{2}\frac{(\vec{r}_{k-1}^T\vec{r}_{k-1})^2}{\vec{r}_{k-1}A\vec{r}_{k-1}} \;\le\; Q(\vec{x}_{k-1}).$$

Ciąg $Q(\vec{x}_k)$ jest nierosnący i ograniczony z dołu przez $Q(\vec{x}^*)$, a więc jest zbieżny. To wymusza

$$\lim_{k\to\infty}\frac{(\vec{r}_{k-1}^T\vec{r}_{k-1})^2}{\vec{r}_{k-1}A\vec{r}_{k-1}} \;=\; 0,$$

co z kolei oznacza, że $\vec{r}_k = \vec{b} - A\vec{x}_k$ zbiega do zera i w konsekwencji

$$\lim_{k\to\infty}\vec{x}_k \;=\; \vec{x}^*.$$

Uwagi i uzupełnienia

U. 11.1 *(Dowód Lematu 11.2)* Jeśli $(\vec{\xi}_j)_{j=1}^n$ jest ortonormalną bazą wektorów własnych macierzy A, to dla dowolnego wektora $\vec{x} = \sum_{j=1}^n \alpha_j\vec{\xi}_j$ mamy

$$\begin{aligned}
\|A\vec{x}\|_2^2 \;&=\; \langle A\vec{x}, A\vec{x}\rangle_2 \;=\; \left\langle \sum_{i=1}^n \alpha_i A(\vec{\xi}_i), \sum_{j=1}^n \alpha_j A(\vec{\xi}_j)\right\rangle_2 \\
&=\; \left\langle \sum_{i=1}^n \alpha_i\lambda_i\vec{\xi}_i, \sum_{j=1}^n \alpha_j\lambda_j\vec{\xi}_j\right\rangle_2 \;=\; \sum_{j=1}^n \alpha_j^2\lambda_j^2 \;\le\; \lambda_1^2\sum_{j=1}^n \alpha_j^2 \\
&=\; \lambda_1^2\|\vec{x}\|_2^2,
\end{aligned}$$

a stąd $\|A\vec{x}\|_2 \le |\lambda_1|\|\vec{x}\|_2$. Z drugiej strony,

$$\|A\vec{\xi}_1\|_2 \;=\; \|\lambda_1\vec{\xi}_1\|_2 \;=\; |\lambda_1|\|\vec{\xi}_1\|_2 \;=\; |\lambda_1|,$$

a więc $\|A\|_2 = |\lambda_1|$.

Aby pokazać drugą część lematu zauważmy, że macierz A^{-1} jest też symetryczna. Ma ona te same wektory własne co A, a wartości własne wynoszą $1/\lambda_j$, przy czym $|1/\lambda_n| \ge \cdots \ge |1/\lambda_1|$. Stąd $\|A^{-1}\|_2 = |1/\lambda_n|$.

U. 11.2 Pokazaliśmy, że warunkiem dostatecznym zbieżności metody iteracji prostej $\vec{x}_k = B\vec{x}_{k-1} + \vec{c}$ jest $\|B\| < 1$. Okazuje się, że można podać warunek konieczny i dostateczny. Metoda iteracji prostej jest zbieżna dla dowolnego przybliżenia początkowego $\vec{x}_0$ wtedy i tylko wtedy, gdy promień spektralny macierzy B jest nie większy od jedności, tzn.

$$\rho(A) = \max |\lambda| < 1,$$

gdzie maksimum wzięte jest po wszystkich wartościach własnych macierzy A.

U. 11.3 Przedstawiając macierz A w postaci $A = L + U$, gdzie L jest macierzą trójkątną dolną elementów występujących na i pod główną przekątną A, układ równań $A\vec{x} = \vec{b}$ jest równoważny układowi $L\vec{x} = -U\vec{x} + \vec{b}$. Stąd wynika metoda iteracji prostej

$$\vec{x}_k = -L^{-1}U\vec{x}_{k-1} + L^{-1}\vec{b},$$

zwana *metodą Seidla*. Można pokazać, że metoda ta jest zbieżna gdy macierz A jest symetryczna i dodatnio określona.

U. 11.4 Pokazana metoda Czebyszewa dla macierzy $A = A^T > 0$ o wartościach własnych z przedziału $[a, b]$, $0 < a < b < \infty$, jest metodą dwupunktową. Rozpatruje się też jednopunktowe, cykliczne metody Czebyszewa postaci

$$\vec{x}_k = \vec{x}_{k-1} + \alpha_k \vec{r}_{k-1},$$

gdzie parametry α_k są wybierane cyklicznie, $\alpha_{j+m} = \alpha_j$ dla wszystkich j i pewnej ustalonej m, oraz tak, aby zminimalizować normę macierzy przejścia w każdym cyklu. Dokładniej, mamy $\vec{x}_k - \vec{x}^* = (I - \alpha_k A)(\vec{x}_{k-1} - \vec{x}^*)$, a stąd

$$\vec{x}_m - \vec{x}^* = W_m(A)(\vec{x}_0 - \vec{x}^*),$$

gdzie

$$W_m(A) = (I - \alpha_m A)(I - \alpha_{m-1}A) \cdots (I - \alpha_1 A).$$

Wielomian $W_m(t)$ jest więc stopnia m, a jego pierwiastki wynoszą $t_j = 1/\alpha_j$, $1 \le j \le m$. Ponadto $W_m(0) = 1$. Jak już wiemy, norma macierzy $\|W_m(A)\|_2$ jest minimalna gdy $W_m(t) = W_m^*(t) = T_m(h(t))/T_m(h(0))$, gdzie $h(t) = (2x - (a + b))/(b - a)$. Stąd parametry α_{j+m} najlepiej jest wybrać tak, aby α_j były odwrotnościami pierwiastków $W_m^*(t)$, czyli

$$\alpha_j = \left(\frac{b+a}{2} - \frac{b-a}{2}\cos\left(\frac{2j-1}{2j}\pi\right)\right)^{-1}, \quad 1 \le j \le m.$$

U. 11.5 Metoda najszybszego spadku należy do rodziny *metod gradientowych* postaci

$$\vec{x}_k = \vec{x}_{k-1} + \alpha_k \vec{r}_{k-1},$$

gdzie parametr α_k wybiera się tak, aby zminimalizować normę

$$\|\vec{x}_k - \vec{x}^*\|_B = \sqrt{(\vec{x}_k - \vec{x}^*)^T B (\vec{x}_k - \vec{x}^*)},$$

a B jest pewną macierzą, $B = B^T > 0$. Zwykle przyjmuje się $B = A^p$. Jak łatwo się przekonać, metodę najszybszego spadku otrzymujemy w przypadku $p = 1$, czyli $B = A$.

Można pokazać, że dla metod gradientowych

$$\|\vec{x}_k - \vec{x}^*\|_B \le \left(\frac{\operatorname{cond}(A) - 1}{\operatorname{cond}(A) + 1}\right)^k \|\vec{x}_0 - \vec{x}^*\|_B.$$

Ćwiczenia

Ćw. 11.1 Niech A będzie macierzą symetryczną o wartościach własnych λ_j, $1 \le j \le n$. Pokazać, że dla dowolnego wielomianu $W(t) = \sum_{j=0}^k a_j t^j$, macierz

$$W(A) = \sum_{j=1}^n a_j A^j$$

jest też symetryczna. Ponadto wektory własne A i $W(A)$ są takie same, a wartości własne $W(A)$ wynoszą $W(\lambda_j)$ dla $1 \le j \le n$.

Ćw. 11.2 Niech $A = A^T > 0$. Rozpatrzmy *metodę Richardsona* postaci

$$\vec{x}_k = (I - \alpha A)\vec{x}_{k-1} + \alpha \vec{b}.$$

gdzie α dobieramy tak, aby zminimalizować normę macierzy $\|I - \alpha A\|_2$. Zauważyć, że jest to metoda iteracji prostej, ale równocześnie jednopunktowa metoda Czebyszewa z U. 11.4 przy $m = 1$. Pokazać, że optymalne α wyraża się wzorem

$$\alpha_{opt} = \frac{2}{\lambda_1 + \lambda_n},$$

gdzie λ_1 i λ_n są odpowiednio największą i najmniejszą wartością własną A. Ponadto

$$\|I - \alpha_{opt} A\|_2 = \frac{\lambda_1 - \lambda_n}{\lambda_1 + \lambda_n} = \frac{\operatorname{cond}(A) - 1}{\operatorname{cond}(A) + 1},$$

gdzie $\operatorname{cond}(A) = \|A\|_2 \|A^{-1}\|_2$.

Ćw. 11.3 Pokazać, że warunkiem koniecznym i dostatecznym zbieżności metody Richardsona z Ćw. 11.2, dla dowolnego przybliżenia początkowego $\vec{x}_0$, jest

$$0 \,<\, \alpha \,<\, \frac{2}{\|A\|_2}. \tag{11.7}$$

Ponadto dla α spełniającego (11.7) iloraz zbieżności wynosi

$$\|I - \alpha A\|_2 \,=\, \begin{cases} \alpha\lambda_1 - 1 & \alpha_{opt} < 2/\lambda_1, \\ 1 - \alpha\lambda_n & 0 < \alpha \le \alpha_{opt}, \end{cases}$$

gdzie $\alpha_{opt} = 2/(\lambda_1 + \lambda_n)$.

Ćw. 11.4 Wykazać, że m-cykliczna jednopunktowa metoda Czebyszewa z U. 11.4 jest zbieżna liniowo z ilorazem równym $(b-a)/(b+a)$ dla $m = 1$ i dążącym do $(\sqrt{b} - \sqrt{a})/(\sqrt{b} + \sqrt{a})$, gdy $m \to \infty$.

Ćw. 11.5 Niech $A = A^T > 0$ i niech

$$Q(\vec{x}) \,=\, \frac{1}{2}\,\vec{x}^T A \vec{x} \,+\, \vec{x}^T \vec{b}.$$

Pokazać, że zbiór

$$\{\, \vec{x} \in \mathbf{R}^n \,:\, Q(\vec{x}) = \text{const.} \,\}$$

tworzy hiperelipsoidę w $\mathbf{R}^n$ o środku $A^{-1}\vec{b}$ i osiach w kierunkach wektorów własnych macierzy A o długościach proporcjonalnych do $1/\sqrt{\lambda_j}$, gdzie λ_j są wartościami własnymi A.
Wskazówka. Wykorzystać Lemat 11.1.

Rozdział 12

Iteracje dla równań nieliniowych

W Rozdziale 11 analizowaliśmy metody iteracyjne dla rozwiązywania równań liniowych $A\vec{x} = \vec{b}$. Teraz zajmiemy się równaniami *nieliniowymi*, w których zamiast operatora liniowego (macierzy) A mamy odwzorowanie nieliniowe $f : D \to \mathbf{R}^n$, gdzie D jest podzbiorem w $\mathbf{R}^n$. Dla uproszczenia będziemy zakładać, że $\vec{b} = \vec{0}$, gdyż w przeciwnym przypadku możemy zastąpić f funkcją $f_1(\vec{x}) = f(\vec{x}) - \vec{b}$. Będziemy również zakładać, że f należy do pewnej znanej klasy F funkcji przekształcających D w $\mathbf{R}^n$ i mających co najmniej jedno zero w D. Zadanie polega więc na znalezieniu rozwiązania $\vec{x}^* = \vec{x}^*(f)$ *równania nieliniowego*

$$f(\vec{x}) = \vec{0}, \qquad \text{dla} \qquad f \in F. \tag{12.1}$$

Jak zwykle w przypadku, gdy danymi są funkcje zakładamy, że jedyną informacją "a priori" o f jest, że $f \in F$. Dodatkową informację możemy zdobyć przez obliczanie wartości, a niekiedy i pochodnych f (o ile istnieją) w wybranych punktach. Zauważmy, że metody iteracyjne Czebyszewa i najszybszego spadku z Rozdziału 11 dla równań liniowych korzystały właśnie z takiej (częściowej) informacji o macierzy A, chociaż formalnie zakładaliśmy, że dostępna jest pełna informacja o jej współczynnikach. Dla równań nieliniowych możliwość obliczenia jedynie informacji częściowej jest założeniem modelowym, a to wymusza już fakt, że każda metoda rozwiązywania równania (12.1) może dawać co najwyżej wynik przybliżony.

12.1 Bisekcja

Metoda bisekcji, czyli *połowienia*, jest dość naturalną metodą obliczania zer skalarnych funkcji ciągłych określonych na danym przedziale $[a, b]$ i zmieniających znak. Dokładniej, rozpatrzmy klasę funkcji

$$F = \{ f \in \mathbf{C}([a, b]) \,:\, f(a) < 0 < f(b) \}. \tag{12.2}$$

Oczywiście, każda funkcja $f \in F$ ma co najmniej jedno zero w $[a, b]$. Startując z przedziału $[a, b]$, w kolejnych krokach metody bisekcji obliczamy informację o wartości f w środku przedziału, co pozwala nam, w zależności od znaku obliczonej wartości, zmniejszyć o połowę przedział, w którym na pewno znajduje się zero funkcji. Bisekcję realizuje następujący ciąg poleceń, po wykonaniu którego x jest przybliżeniem zera funkcji f z zadaną dokładnością ε.

```
xl := a;   xr := b;
x := (a + b)/2;   e := (b − a)/2;
while  (e > ε)  do
begin
     if  (f(x) < 0)  then  xr := x  else  xl := x;
     x = (xl + xr)/2;   e := e/2
end.
```

Z konstrukcji metody łatwo wynika, że po wykonaniu k iteracji (albo po obliczeniu k wartości funkcji) otrzymujemy $x = x_k$, które odległe jest od pewnego rozwiązania x^* o co najwyżej

$$|x_k - x^*| \,\le\, \Big(\frac{1}{2}\Big)^k \Big(\frac{b-a}{2}\Big). \tag{12.3}$$

Metoda bisekcji jest więc zbieżna *liniowo* z ilorazem $1/2$. Choć ta zbieżność nie jest imponująca, bisekcja ma kilka istotnych zalet. Oprócz jej prostoty, należy podkreślić fakt, że bisekcja jest w pewnym sensie uniwersalna. Dla jej zbieżności wystarcza bowiem jedynie ciągłośc funkcji. Poza tym możemy łatwo kontrolować jej błąd. Konsekwencją (12.3) jest bowiem następujący wniosek.

Wniosek 12.1 *Dla znalezienia zera z dokładnością $\varepsilon > 0$, wystarczy obliczyć w metodzie bisekcji*

$$k = k(\varepsilon) = \left\lceil \log_2 \frac{(b-a)}{\varepsilon} \right\rceil - 1$$

wartości funkcji.

Dodajmy jeszcze, że bisekcja minimalizuje błąd najgorszy w klasie F zdefiniowanej przez (12.2), wśród wszystkich algorytmów korzystających z określonej liczby obliczeń wartości funkcji, zob U. 12.1.

12.2 Iteracje proste

Przedstawimy teraz metodę *iteracji prostych* dla rozwiązywania równań nieliniowych, którą można traktować jako uogólnienie iteracji prostych dla równań liniowych. Najpierw równanie (12.1) przekształcamy do równania równoważnego (tzn. mającego te same rozwiązania)

$$\vec{x} = \phi(\vec{x}). \tag{12.4}$$

Następnie, startując z pewnego przybliżenia początkowego $\vec{x}_0$, konstruujemy ciąg kolejnych przybliżeń $\vec{x}_k$ według wzoru

$$\vec{x}_k = \phi(\vec{x}_{k-1}), \qquad k \geq 1.$$

Twierdzenie 12.1 *Niech D_0 będzie domkniętym podzbiorem dziedziny D,*

$$\overline{D}_0 = D_0 \subset D,$$

w którym ϕ jest odwzorowaniem zwężającym. To znaczy, $\phi(D_0) \subset D_0$, oraz istnieje stała $0 \leq L < 1$ taka, że

$$\|\phi(\vec{x}) - \phi(\vec{y})\| \leq L \, \|\vec{x} - \vec{y}\|, \qquad \forall \vec{x}, \vec{y} \in D_0.$$

Wtedy równanie (12.4) ma dokładnie jedno rozwiązanie $\vec{x}^*$, oraz

$$\vec{x}^* = \lim_{k \to \infty} \vec{x}_k,$$

dla dowolnego przybliżenia początkowego $\vec{x}_0 \in D_0$.

Dowód Wobec

$$\begin{aligned} \|\vec{x}_k - \vec{x}_{k-1}\| &= \|\phi(\vec{x}_{k-1}) - \phi(\vec{x}_{k-2})\| \le L\,\|\vec{x}_{k-1} - \vec{x}_{k-2}\| \\ &\le \cdots \le L^{k-1}\|\vec{x}_1 - \vec{x}_0\|, \end{aligned}$$

dla $k \ge s$ mamy

$$\begin{aligned} \|\vec{x}_k - \vec{x}_s\| &\le \sum_{j=s+1}^{k} \|\vec{x}_j - \vec{x}_{j-1}\| \le \sum_{j=s+1}^{k} L^{j-1}\|\vec{x}_1 - \vec{x}_0\| \\ &= L^s(1 + L + \cdots + L^{k-s-1})\|\vec{x}_1 - \vec{x}_0\| \le \frac{L^s}{1-L}\|\vec{x}_1 - \vec{x}_0\|. \end{aligned}$$

Ciąg $\{\vec{x}_k\}_k$ jest więc ciągiem Couchy'ego. Stąd istnieje granica $\vec{\alpha} = \lim_{k\to\infty} \vec{x}_k$, która należy do D_0, wobec domkniętości tego zbioru. Ponieważ lipschitzowskość ϕ implikuje jej ciągłość, mamy też

$$\phi(\vec{\alpha}) = \phi\Big(\lim_{k\to\infty} \vec{x}_k\Big) = \lim_{k\to\infty} \phi(\vec{x}_k) = \lim_{k\to\infty} \vec{x}_k = \vec{\alpha},$$

tzn. $\vec{\alpha}$ jest punktem stałym odwzorowania ϕ. Dla jednoznaczności zauważmy, że jeśliby istniał drugi, różny od $\vec{\alpha}$, punkt stały $\vec{\beta}$, to mielibyśmy

$$\|\vec{\alpha} - \vec{\beta}\| = \|\phi(\vec{\alpha}) - \phi(\vec{\beta})\| \le L\,\|\vec{\alpha} - \vec{\beta}\|.$$

Stąd $1 < L$, co jest sprzeczne z założeniem, że ϕ jest zwężająca. □

Z powyższych rozważań otrzymujemy natychmiastowy wniosek dotyczący zbieżności iteracji prostych.

Wniosek 12.2 *Przy założeniach twierdzenia 12.1, metoda iteracji prostych jest zbieżna co najmniej liniowo z ilorazem L, tzn.*

$$\|\vec{x}_k - \vec{x}^*\| \le L^k\,\|\vec{x}_0 - \vec{x}^*\|.$$

Dla ilustracji, rozpatrzmy natępujące proste równanie skalarne:

$$x = \cos(x), \qquad \text{dla} \qquad x \in D = \mathbf{R}. \tag{12.5}$$

W tym przypadku $\phi(x) = \cos(x)$. Zauważamy, że w przedziale $[0, 1]$ funkcja ϕ jest zwężająca ze stałą

$$L = \max_{0\le x\le 1} |\cos'(x)| = \sin(1) < 1.$$

Stąd istnieje dokładnie jedno rozwiązanie równania (12.5) w przedziale $[0,1]$. Rozwiązanie to może być aproksymowane z dowolnie małym błędem przy pomocy iteracji prostych, startując z dowolnego przybliżenia początkowego $\vec{x}_0 \in [0,1]$.

Zaletą iteracji prostych jest fakt, że zbieżność nie zależy od wymiaru n zadania, ale tylko od stałej Lipschitza L. Ma ona szczególne zastosowanie w przypadku, gdy funkcja ϕ jest zwężająca na całym zbiorze D, tzn. $D_0 = D$. Jeśli ponadto D ma skończoną średnicę $\mathrm{diam}(D)$, to dla osiągnięcia ε-sproksymacji zera funkcji f wystarczy wykonać

$$k \,=\, k(\varepsilon) \,=\, \left\lceil \frac{\log(\|\vec{x}_0 - \vec{x}^*\|/\varepsilon)}{\log(1/L)} \right\rceil \,=\, \left\lceil \frac{\log(\mathrm{diam}(D)/\varepsilon)}{\log(1/L)} \right\rceil$$

iteracji, niezależnie od x_0. Metody zbieżne dla dowolnego przybliżenia początkowego, nazywamy *zbieżnymi globalnie*.

12.3 Metody interpolacyjne

Metody iterpolacyjne dla równań nieliniowych powstają podobnie jak kwadratury interpolacyjne dla całkowania. Najpierw daną funkcję interpolujemy wielomianem Lagrange'a albo Hermite'a ustalonego stopnia, a następnie jako przybliżenie zera bierzemy zero tego wielomianu iterpolacyjnego. Proces taki można iterować (powtarzać) wymieniając za każdym razem jeden z punktów interpolacyjnych na punkt nowo obliczony. Zauważmy, że w ogólności takie metody muszą korzystać w kolejnych iteracjach nie tylko z ostatniego przybliżenia x_{k-1}, ale również z kilku poprzednich, tzn.

$$x_k \,=\, \phi(x_{k-1}, \ldots, x_{k-s}).$$

Wymagają one również podania nie jednego, ale s przybliżeń początkowych $x_0, \ldots, x_{s-1}$.

W praktyce stosuje się jednak tylko interpolację wielomianami niskich stopni, np. wielomianami liniowymi lub najwyżej kwadratowymi, których zera można łatwo obliczyć korzystając z bezpośrednich formuł.

Jak się przekonamy, metody interpolacyjne należą do grupy metod *zbieżnych lokalnie*. Znaczy to, że zbieżność ciągu $\{x_k\}_k$ do zera danej

funkcji f jest zapewniona jedynie wtedy, gdy przybliżenia początkowe zostały wybrane dostatecznie blisko x^*.

W dalszych rozważaniach będziemy zakładać dla uproszczenia, że dziedzina $D = \mathbf{R}$.

W przypadku, gdy wielomianem interpolacyjnym jest wielomian liniowy Hermite'a, mówimy o *metodzie Newtona*, albo *stycznych*. Startując z pewnego przybliżenia początkowego x_0, w kolejnych krokach metody, k-te przybliżenie x_k jest punktem przecięcia stycznej do wykresu f w punkcie x_{k-1}. Ponieważ równanie stycznej wynosi $y(x) = f(x_{k-1}) + f'(x_{k-1})(x - x_{k-1})$, otrzymujemy wzór

$$x_k \;=\; x_{k-1} \;-\; \frac{f(x_{k-1})}{f'(x_{k-1})}.$$

Oczywiście, aby metoda Newtona była dobrze zdefiniowana, musimy założyć, że $f'(x_{k-1})$ istnieje i nie jest zerem.

Zauważmy, że metodę Newtona można traktować jako szczególny przypadek iteracji prostych, gdzie

$$\phi(x) \;=\; x - \frac{f(x)}{f'(x)}.$$

Widać też, że nie jest ona zbieżna globalnie. Nawet jeśli pochodna w x_{k-1} się nie zeruje, ciąg $\{x_k\}_k$ może nie zbiegać do zera funkcji f, zob. Ćw. 12.2. Okazuje się jednak, że jeśli wystartujemy dostatecznie blisko rozwiązania x^*, to metoda Newtona jest zbieżna. Dokładniej, załóżmy najpierw, że $f(x^*) = 0$ oraz

$$f'(x^*) \;\neq\; 0.$$

Ponadto załóżmy, że f jest dwukrotnie różniczkowalna w sposób ciągły, $f \in \mathbf{C}^2(D)$. Rozwijając ϕ w szereg Taylora w punkcie x^* otrzymujemy

$$x_k - x^* \;=\; \phi(x_{k-1}) - \phi(x^*) \;=\; (x_{k-1} - x^*)\phi'(x^*) + (x_{k-1} - x^*)^2\phi''(\xi_k)/2,$$

gdzie $\min(x^*, x_{k-1}) \;\leq\; \xi_k \;\leq\; \max(x^*, x_{k-1})$. Wobec tego, że $\phi'(x^*) = f(x)f''(x)/(f'(x))^2 = 0$ i $\phi''(\xi_k) = f''(\xi_k)/f'(\xi_k)$, mamy

$$x_k - x^* \;=\; (x_{k-1} - x^*)^2 \frac{f''(\xi_k)}{2f'(\xi_k)}. \qquad (12.6)$$

Zdefiniujmy liczbę

$$R_f \;=\; \sup_{r\ge 0}\;\sup_{\{x:|x-x^*|\le r\}} \Big|\frac{2(x-x^*)f''(x)}{f'(x)}\Big| \;<\; 1.$$

Oczywiście $R_f > 0$. Dla x_{k-1} spełniającego $|x_{k-1} - x^*| \le R < R_f$, mamy z równości (12.6)

$$|x_k - x^*| \;\le\; q\,|x_{k-1} - x^*|,$$

gdzie $q < 1$ i q zależy tylko od R.

Niech teraz x^* będzie zerem m-krotnym,

$$f(x^*) = f'(x^*) = \cdots = f^{(m-1)}(x^*) = 0 \neq f^{(m)}(x^*),$$

gdzie $m \ge 2$, oraz niech f będzie m-krotnie różniczkowalna w sposób ciągły. Wtedy

$$\begin{aligned} x_k - x^* &= (x_{k-1} - x^*) - \frac{(x_{k-1}-x^*)^m f^{(m)}(\eta_k^{(1)})/m!}{(x_{k-1}-x^*)^{m-1} f^{(m-1)}(\eta_k^{(2)})/(m-1)!} \\ &= (x_{k-1} - x^*)\left(1 - \frac{1}{m}\frac{f^{(m)}(\eta_k^{(1)})}{f^{(m)}(\eta_k^{(2)})}\right) \\ &\approx (x_{k-1} - x^*)\Big(1 - \frac{1}{m}\Big), \end{aligned} \tag{12.7}$$

o ile x_{k-1} jest “blisko” x^*.

Metoda Newtona jest więc zbieżna lokalnie. Z (12.6) i (12.7) można też wywnioskować, jaki jest charakter zbieżności metody Newtona. Dla zera jednokrotnego x^* oraz $f''(x^*) \neq 0$ mamy bowiem

$$(x_k - x^*) \;\approx\; (x - x_{k-1})^2 \frac{f''(x^*)}{2f'(x^*)}.$$

Mówimy, że zbieżność jest *kwadratowa*. Jeśli zaś $f''(x^*) = 0$ to zbieżnosc jest nawet szybsza. Z kolei dla zera m-krotnego zbieżność jest liniowa z ilorazem $(1 - 1/m)$.

Metoda Newtona jest pierwszą poznaną tutaj metodą iteracyjną, która jest (dla zer jednokrotnych) zbieżna szybciej niż liniowo. Dla takich metod wprowadza się pojęcie *wykładnika zbieżności*, który jest zdefiniowany następująco.

Powiemy, że metoda iteracyjna ϕ jest w klasie funkcji F rzędu co najmniej $p \geq 1$, gdy spełniony jest następujący warunek. Niech $f \in F$ i $f(x^*) = 0$. Wtedy istnieje stała $C < \infty$ taka, że dla dowolnych przybliżeń początkowych $x_0, \ldots, x_{s-1}$ dostatecznie bliskich x^*, kolejne przybliżenia $x_k = \phi(x_{k-1}, \ldots, x_{k-s})$ generowane tą metodą spełniają

$$|x_k - x^*| \leq C\, |x_{k-1} - x^*|^p.$$

Ponadto, jeśli $p = 1$ to dodatkowo żąda się, aby $C < 1$.

Definicja 12.1 *Wykładnikiem zbieżności metody iteracyjnej ϕ w klasie F nazywamy liczbę p^* zdefiniowaną równością*

$$p^* = \sup\{\, p \geq 1 : \phi \text{ jest rzędu co najmniej } p \,\}.$$

Możemy teraz sformułować następujące twierdzenie, które natychmiast wynika z poprzednich rozważań.

Twierdzenie 12.2 *Wykładnik zbieżności metody Newtona (stycznych) wynosi $p^* = 2$ w klasie funkcji o zerach jednokrotnych, oraz $p^* = 1$ w klasie funkcji o zerach wielokrotnych.*

Inną znaną i często używaną metodą interpolacyjną jest *metoda siecznych*, w której stosuje się liniowy wielomian interpolacyjny Lagrange'a. Metoda ta wykorzystuje więc do konstrukcji x_k przybliżenia x_{k-1} i x_{k-2}. Musimy również wybrać dwa różne punkty startowe x_0 i x_1. Ponieważ prosta interpolująca f w x_{k-1} i x_{k-2} ma wzór

$$y(x) = \frac{x - x_{k-2}}{x_{k-1} - x_{k-2}} f(x_{k-1}) + \frac{x - x_{k-1}}{x_{k-2} - x_{k-1}} f(x_{k-2}),$$

otrzymujemy

$$x_k = x_{k-1} - \frac{x_{k-1} - x_{k-2}}{f(x_{k-1}) - f(x_{k-2})}\, f(x_{k-1}).$$

Zauważmy, że jeśli x_{k-1} i x_{k-2} są blisko siebie, to x_k jest podobny do tego z metody Newtona, bowiem wtedy iloraz różnicowy $(f(x_{k-1}) -$

$f(x_{k-2}))/(x_{k-1} - x_{k-2})$ dobrze przybliża pochodną $f'(x_{k-1})$. Nie wystarcza to jednak, aby osiągnąć zbieżność z wykładnikiem 2. Dokładniej, można pokazać, że wykładnik zbieżności metody siecznych dla zer jednokrotnych wynosi $p^* = (1 + \sqrt{5})/2 = 1.618\ldots$, zob. U. 12.3.

Niewątpliwą zaletą metody siecznych jest jednak to, że nie wymaga ona obliczania pochodnej funkcji (co w praktyce jest często bardzo trudne, a niekiedy nawet niemożliwe), a tylko jej wartości.

12.4 Obliczanie zer wielomianów

Obliczanie zer w przypadku, gdy funkcja f jest wielomianem jest właściwie osobnym zadaniem–my tylko dotkniemy tego tematu.

Załóżmy, że wielomian $w : \mathbf{R} \to \mathbf{R}$ jest dany poprzez jego współczynniki a_j w bazie potęgowej,

$$w(x) = \sum_{j=0}^{n} a_j x^j$$

($a_n \neq 0$), i zastosujmy metodę Newtona dla znalezienia największego zera tego wielomianu. Jedna iteracja metody Newtona jest wtedy stosunkowo prosta, gdyż obliczenie wartości i pochodnej wielomianu w kolejnym pukcie $\xi = x_{k-1}$ polega po prostu na dwukrotnym zastosowaniu schematu Hornera. Rzeczywiście, przypomnijmy, że schemat Hornera oblicza nie tylko wartość $w(\xi)$, ale również współczynniki w bazie potęgowej ilorazu $w_1(x)$ przy dzieleniu $w(x)$ przez jednomian $(x - \xi)$, zob. Ćw. 6.2. Zauważmy dalej, że równość

$$w(x) = (x - \xi)w_1(x) + w(\xi)$$

implikuje $w'(\xi) = w_1(\xi)$. Stąd schemat obliczania jednocześnie $w(\xi)$ i $w'(\xi)$ wygląda następująco.

```
w0 := w1 := a_n;
for j := n − 1 downto 1 do
begin
    w0 := w0 ∗ ξ + a_j;
    w1 := w1 ∗ ξ + w0
end;
w0 := w0 + a_0.
```

Po wykonaniu tych poleceń mamy $w0 = w(\xi)$ i $w1 = w'(\xi)$.

Twierdzenie 12.3 *Załóżmy, że wielomian $w(x) = \sum_{j=0}^{n} a_j x^j$ ma zera rzeczywiste*

$$\xi_1 \geq \xi_2 \geq \cdots \geq \xi_n.$$

Jeśli tylko przybliżenie początkowe $x_0 > \xi_1$, to ciąg x_k generowany metodą Newtona jest ściśle malejący i zbieżny do ξ_1.

Dowód Wielomian w można przedstawić w postaci

$$w(x) \;=\; (x-\xi_1)w_1(x),$$

gdzie $w_1(x) = a_n(x-\xi_2)\cdots(x-\xi_n)$. Wtedy

$$w'(x) \;=\; w_1(x) + (x-\xi_1)w_1'(x),$$

a stąd

$$\begin{aligned} x_k - \xi_1 \;&=\; (x_{k-1}-\xi_1) \;-\; \frac{w(x_{k-1})}{w'(x_{k-1})} \qquad (12.8)\\ &=\; (x_{k-1}-\xi_1)\left(1 - \frac{w_1(x_{k-1})}{w_1(x_{k-1}) + (x_{k-1}-\xi_1)w_1'(x_{k-1})}\right). \end{aligned}$$

Zauważmy teraz, że wielomiany $w_1(x)$ i $w_1'(x)$ mają stały znak dla $x > \xi_1$, który równy jest znakowi a_n. Rzeczywiście, ponieważ $w_1(x)$ ma wszystkie $n-1$ pierwiastków w przedziale $(-\infty, \xi_1]$, to $w_1'(x)$ ma również wszystkie $n-2$ pierwiastki w tym samym przedziale. Stąd oba wielomiany mają ten sam znak na $(\xi_1, +\infty)$ równy znakowi współczynnika przy najwyższych potęgach tych wielomianów. Współczynnik ten jest w obu przypadkach równy a_n.

Z rozważań tych wynika, że mnożnik przy $(x_{k-1} - \xi_1)$ w prawej stronie równania (12.8) jest dla $x_{k-1} > \xi_1$ dodatni i mniejszy od 1, a stąd, że ciąg x_k jest malejący i ograniczony z dołu przez ξ_1. Niech $g = \lim_{k\to\infty} x_k \geq \xi_1$. Biorąc granicę obu stron (12.8) przy $k \to \infty$ otrzymujemy

$$g - \xi_1 \;=\; (g-\xi_1)\left(1 - \frac{w_1(g)}{w_1(g) + (g-\xi_1)w_1'(g)}\right),$$

co wymusza $g = \xi_1$, kończąc dowód. $\square$

Dodajmy, że jeśli zero ξ_1 jest jednokrotne, $\xi_1 > \xi_2$, to mamy zbieżność kwadratową. Należy jednak podkreślić, że zbieżność kwadratową uzyskujemy dopiero pod koniec procesu iteracyjnego. Mamy bowiem

$$\lim_{x\to+\infty} \frac{w_1(x)}{w_1(x) + (x-\xi_1)w_1'(x)} \;=\; \lim_{x\to+\infty} \Big(1 + \frac{xw_1'(x)}{w_1(x)}\Big)^{-1} \;=\; \frac{1}{n}.$$

Jeśli więc wystartujemy z $x_0 \gg \xi_1$ to początkowo zaobserwujemy zbieżność tylko liniową z ilorazem $1-1/n$. W miarę zbliżania się do ξ_1 zbieżność będzie rosła, aż osiągnie w końcu poziom zbieżności kwadratowej,

$$x_k - \xi_1 \;\approx\; (x_{k-1} - \xi_1)^2 \frac{w_1'(\xi_1)}{w_1(\xi_1)},$$

Dla zera wielokrotnego, $\xi_1 = \cdots = \xi_m > \xi_{m+1}$, $m < n$, zbieżność pozostanie jednak cały czas na poziomie liniowym, przy czym iloraz zbieżności wzrośnie do $1 - 1/m$. W końcu, jeśli ξ_1 jest zerem o maksymalnej krotności n, to łatwo zauważyć, że

$$x_k - \xi_1 \;=\; \Big(1 - \frac{1}{n}\Big)(x_{k-1} - \xi_1).$$

Pozostaje jeszcze problem znalezienia przybliżenia $x_0 > \xi_1$. Można to zrobić na kilka sposobów. Na przykład, zauważmy, że dla

$$x \;>\; M \;=\; \max\Big(1, \sum_{j=0}^{n-1} \frac{|a_j|}{|a_n|}\Big)$$

mamy

$$\frac{w(x)}{a_n} \;\ge\; x^n\Big(1 \;-\; \sum_{j=0}^{n-1} \frac{|a_j|}{|a_n x^{n-j}|}\Big) \;\ge\; x^n\Big(1 \;-\; \frac{1}{M}\sum_{j=0}^{n-1} \frac{|a_j|}{|a_n|}\Big) \;>\; 0.$$

Stąd $w(x)$ ma stały znak na $(M, +\infty)$, a to oznacza, że $\xi_1 \le M$. Jako x_0 można więc wziąć dowolną liczbę większą od M.

Uwagi i uzupełnienia

U. 12.1 Metoda bisekcji jest optymalna w następującym sensie. Niech $A : F \to \mathbf{R}$ będzie dowolną metodą (algorytmem) aproksymującą zero $x^*(f)$ funkcji f z klasy F zdefiniowanej w (12.2), korzystającą jedynie z obliczeń (informacji o) f w k punktach. Wtedy dla dowolnego $\gamma > 0$ istnieje funkcja $f_\gamma \in F$ mająca tylko jedno zero $x^*(f_\gamma)$ w $[a, b]$ i taka, że

$$|A(f_\gamma) - x^*(f_\gamma)| \geq \left(\frac{1}{2}\right)^k \left(\frac{b-a}{2}\right) + \gamma.$$

Co więcej, można pokazać, że fakt ten zachodzi też w węższej klasie F_1 funkcji $f \in F$, które są dowolnie wiele razy różniczkowalne. Oczywiście, nie wyklucza to istnienia metod iteracyjnych (takich jak metoda Newtona), które dla $f \in F_1$ są zbieżne szybciej niż liniowo.

U. 12.2 Wszystkie metody iteracyjne opisane w tym rozdziale należą do grupy *metod adaptacyjnych*. Są to metody, które obliczają kolejną wartość (ew. pochodną) funkcji f w punkcie x_k, którego wybór istotnie zależy od poprzednio obliczonych wartości f, tzn. $x_k = x_k(f(x_0), \ldots, f(x_{k-1}))$. Okazuje się, że metody nieadaptacyjne są dużo gorsze przy rozwiązywaniu równań nieliniowych. Na przykład, najlepsza metoda korzystająca z k wartości funkcji wybranych nieadaptacyjnie ma w klasie (12.2) błąd najgorszy równy tylko $(b - a)/(2k + 2)$.

Zauważmy jeszcze, że dla zadania całkowania z Rozdziałów 9 i 10, optymalne metody okazały się nieadaptacyjne.

U. 12.3 Uzasadnimy teraz, że dla zer jednokrotnych wykładnik zbieżności metody siecznych wynosi $p^* = (1 + \sqrt{5})/2$.

Niech $f \in \mathbf{C}^2(D)$. Wykorzystując ilorazy różnicowe otrzymujemy

$$\begin{aligned}
x_k - x^* &= (x_{k-1} - x^*) - \frac{f(x_{k-1})}{f(x_{k-2}, x_{k-1})} \\
&= (x_{k-1} - x^*)\left(1 - \frac{f(x_{k-1}, x^*)}{f(x_{k-2}, x_{k-1})}\right) \\
&= (x_{k-1} - x^*)(x_{k-2} - x^*)\frac{f(x_{k-2}, x_{k-1}, x^*)}{f(x_{k-2}, x_{k-1})} \\
&= (x_{k-1} - x^*)(x_{k-2} - x^*)\frac{f''(\xi_k^{(1)})}{2f'(\xi_k^{(2)})}.
\end{aligned}$$

Stąd widać, że dla x_{k-2} i x_{k-1} dostatecznie bliskich x^*, ciąg x_k zbiega do x^*. Aby znaleźć wykładnik zbieżności, załóżmy bez zmniejszenia ogólności, że $f''(x^*) \neq 0$ (w przeciwnym przypadku zbieżność jest jeszcze szybsza). Wtedy

$$x_k - x^* \approx C(x_{k-1} - x^*)(x_{k-2} - x^*), \tag{12.9}$$

gdzie $C = f''(x^*)/(2f'(x^*))$. Stąd, po podstawieniu $z_j = \ln(|C(x_j - x^*)|)$, dostajemy

$$z_k \approx z_{k-1} + z_{k-2}.$$

Ostatnie równanie jest *równaniem różnicowym*, którego rozwiązaniem jest $z_k \approx q^k$, gdzie $q = (1 + \sqrt{5})/2$ jest dominującym pierwiastkiem równania $z^2 = z + 1$. Ostatecznie więc $|x_k - x^*| \approx |C^{-1} e^{q^k}|$, czyli

$$|x_k - x^*| \approx |C|^{q-1}\Big(|C^{-1} e^{q^{k-1}}\Big)^q \approx |C|^{q-1}|x_{k-1} - x^*|^q.$$

U. 12.4 Dla znalezienia zera funkcji $f : \mathbf{R}^n \to \mathbf{R}^n$ można zastosować wielowymiarową metodę Newtona. Jest ona zdefiniowana następującym wzorem rekurencyjnym:

$$\vec{x}_k = \vec{x}_{k-1} - (f'(\vec{x}_{k-1}))^{-1} f(\vec{x}_{k-1}),$$

gdzie $f'(\vec{\xi})$ jest macierzą–*Jacobianem* funkcji f wziętym w punkcie $\vec{\xi}$, tzn.

$$f'(\vec{\xi}) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1}(\vec{\xi}) & \cdots & \frac{\partial f_1}{\partial x_n}(\vec{\xi}) \\ \vdots & & \vdots \\ \frac{\partial f_n}{\partial x_1}(\vec{\xi}) & \cdots & \frac{\partial f_n}{\partial x_n}(\vec{\xi}) \end{pmatrix},$$

$$f(\vec{\xi}) = \begin{pmatrix} f_1(\vec{\xi}) \\ \vdots \\ f_n(\vec{\xi}) \end{pmatrix}, \qquad \vec{\xi} = \begin{pmatrix} \xi_1 \\ \vdots \\ \xi_n \end{pmatrix}.$$

W praktyce metodę tą realizuje się rozwiązując w każdym kroku iteracyjnym układ równań liniowych $n \times n$. Dokładniej,

$$\vec{x}_k = \vec{x}_{k-1} - \vec{y}_{k-1},$$

gdzie $\vec{y}_{k-1}$ jest rozwiązaniem układu

$$f'(\vec{x}_{k-1})\vec{y}_{k-1} = f(\vec{x}_{k-1}).$$

Tak jak w przypadku jednowymiarowym, wielowymiarowa metoda Newtona jest zbieżna lokalnie do zera $\vec{x}^*$ z wykładnikiem 2, o ile Jacobian w punkcie $\vec{x}^*$ jest nieosobliwy, tzn. $\det(f'(\vec{x}^*)) \neq 0$.

Ćwiczenia

Ćw. 12.1 Uzasadnić, że metoda iteracji prostych $x_k = \cos(x_{k-1})$ jest zbieżna do jedynego rozwiązania równnania $x = \cos(x)$ dla dowolnego przybliżenia początkowego $x_0 \in \mathbf{R}$.

Ćw. 12.2 Podać przykład funkcji f i przybliżenia początkowego x_0 takich, że ciąg generowany metodą Newtona oscyluje, tzn. $x_{2s} = x_0$ i $x_{2s+1} = x_1$, dla wszystkich s.

Ćw. 12.3 Jeśli w modelu obliczeniowym liczenie pierwiastka kwadratowego $\sqrt{a}$ $(a > 0)$ nie jest dopuszczalne, to możemy go przybliżyć stosując wzór rekurencyjny

$$x_k = \frac{1}{2}\left(x_{k-1} + \frac{a}{x_{k-1}}\right),$$

który powstaje przez zastosowanie metody Newtona do równania $f(x) = x^2 - a = 0$. Pokazać, że ciąg x_k jest zbieżny do $\sqrt{a}$ dla dowolnego przybliżenia początkowego $x_0 > 0$. Scharakteryzować szybkość zbieżności.

Ćw. 12.4 Zaproponować metodę iteracyjną obliczania $1/a$ dla $a > 0$ nie używającą dzielenia. Jak wybrać przybliżenie początkowe aby metoda była zbieżna? Jaki jest wykładnik zbieżności?
Wskazówka. Zastosować metodę Newtona do funkcji $f(x) = 1/x - a$.

Ćw. 12.5 Rozpatrzmy metodę iteracyjną

$$x_k = x_{k-1} - \frac{x_{k-1} - a}{f(x_{k-1}) - f(a)} f(x_{k-1}),$$

gdzie a jest pewnym parametrem. Pokazać, że metoda ta jest zbieżna liniowo, o ile a jest dostatecznie blisko zera x^*.

Ćw. 12.6 Pokazać, że *metoda Steffensena*

$$x_k = x_{k-1} - \frac{f^2(x_{k-1})}{f(x_{k-1} + f(x_{k-1})) - f(x_{k-1})}$$

jest zbieżna lokalnie z wykładnikiem 2.

Ćw. 12.7 Rozpatrzmy metodę iteracyjną daną wzorem

$$x_k = x_{k-1} - m\frac{f(x_{k-1})}{f'(x_{k-1})}.$$

Pokazać, że jeśli f ma m-krotne zero x^* to metoda ta jest lokalnie zbieżna do x^* z wykładnikiem 2.